Evaluating Cognitive Maps and Planning in Large Language Models with CogEval

Ida Momennejad[*] Microsoft Research New York, NY idamo

Hosein Hasanbeig[*] Microsoft Research New York, NY hosein.hasanbeig

Felipe Vieira Frujeri[*] Microsoft Redmond, WA felipe.frujeri

Hiteshi Sharma Microsoft Redmond, WA hiteshi.sharma

Robert Osazuwa Ness Microsoft Research Redmond, WA robertness

Nebojsa Jojic Microsoft Research Redmond, WA jojic

Hamid Palangi Microsoft Research Redmond, WA hpalangi

Jonathan Larson Microsoft Research Redmond, WA jolarso

@microsoft.com

Abstract

Recently an influx of studies claim emergent cognitive abilities in large language models (LLMs). Yet, most rely on anecdotes, overlook contamination of training sets, or lack systematic Evaluation involving multiple tasks, control conditions, multiple iterations, and statistical robustness tests. Here we make two major contributions. First, we propose CogEval, a cognitive science-inspired protocol for the systematic evaluation of cognitive capacities in Large Language Models. The CogEval protocol can be followed for the evaluation of various abilities. Second, here we follow CogEval to systematically evaluate *cognitive maps* and *planning ability* across eight LLMs (OpenAI GPT-4, GPT-3.5-turbo-175B, davinci-003-175B, Google Bard, Cohere-xlarge-52.4B, Anthropic Claude-1-52B, LLaMA-13B, and Alpaca-7B). We base our task prompts on human experiments, which offer both established construct validity for evaluating planning, and are absent from LLM training sets. We find that, while LLMs show apparent competence in a few planning tasks with simpler structures, systematic evaluation reveals striking failure modes in planning tasks, including hallucinations of invalid trajectories and getting trapped in loops. These findings do not support the idea of emergent out-of-the-box planning ability in LLMs. This could be because LLMs do not understand the latent relational structures underlying planning problems, known as cognitive maps, and fail at unrolling goal-directed trajectories based on the underlying structure. Implications for application and future directions are discussed.

1 Introduction

Large language models (LLMs) are generatively pre-trained and display apparent competence on some cognitive tasks [9]. This has led to a recent surge in studies claiming LLMs have emergent human-level cognitive abilities, which may encourage applications that interact with LLMs in a zero-shot or few-shot manner with expectations of human-level cognition. However, most claims of competence are based on anecdotes rather than systematic evaluation. In response, we make two contributions. First, we propose CogEval, a Cognitive Science-Inspired [14, 6, 47] protocol for Measurement and Evaluation of cognitive abilities in LLMs (Figure 1, top), such as planning, theory

[*]Equal contribution

Preprint. Under review.

of mind, causal inference, or other abilities. Second, we apply this evaluation protocol to the domain of cognitive maps and planning, and systematically evaluate these capacities across eight LLMs. We build our task prompts according to established human experiments, but our goal is not a comparison with human performance nor any assumptions of LLMs being "human-like" [34]. We evaluate LLMs' *functional* as opposed to *formal linguistic* abilities [27], and by that we have both a functionalist and multiple-realizability-based notion of cognitive ability [10] in mind.

We investigated whether LLMs (OpenAI GPT-4, GPT-3.5-175B, and davinci-003-175B, Google Bard, Cohere-52.4B, Anthropic Claude-1-52B, LLaMA-13B, and Alpaca-7B) understand the latent structure of planning problems (cognitive maps). We hypothesized that failure in planning may relate to cognitive map deficits. To address these questions, we followed the CogEval protocol (Figure 1). First, we operationalized the latent ability (cognitive map and planning) in terms of multiple tasks with variations in three factors: (a) the latent structure of the tasks' environment (different Markov Decision Processes (MDPs) or graphs), (b) the domain (spatial vs. social ties vs. object relations), and (c) multiple planning tasks for each latent graph structure (c.f. Methods for detail). These domains were selected due to their prevalence in everyday problems as well as the cognitive science literature on cognitive maps [5, 53, 43]. We then generated repeated measurements across small and large LLMs (c.f. Methods for choice of LLMs) and conducted statistical analysis to compare the results. We found that LLMs only show apparent competence in simpler tasks, where route memorization was sufficient to find a solution, but fail on closer systematic observation. Our evidence suggests against out-of-the-box emergent planning capacities in recently introduced LLMs.

What is a cognitive map? A cognitive map is a representation of latent relational structures that underlies a task or environment, and facilitates planning, reasoning, and inference in biological and artificial problems [55, 5, 33, 8]. The concept originated from Tolman's latent learning experiments, demonstrating rodents' ability to learn maze structures without rewards [55]. This challenged the dominant behaviorist view that learning only occurs with reinforcement; and paved the way for a cognitive revolution. Decades later, discoveries of hippocampal place cells [39, 38, 40] and entorhinal cortex grid cells [15, 17, 36], together referred to as "the brain's GPS," further substantiated cognitive maps and earned the 2014 Nobel Prize [1]. Cognitive maps have since been studied behaviorally, computationally, and neurally; revealing that multi-step, multi-scale, and compressed neural representations are crucial for inference in both memory and planning [5, 33, 8]. Over the past decades, a number of Reinforcement Learning (RL) and deep neural network models have been proposed to capture the computations involved in cognitive maps and planning in the hippocampus and the prefrontal cortex of humans, rodents, bats, monkeys, and birds [5, 45, 8].

Why would LLMs plan with a cognitive map? It has been suggested that the transformer architecture and its learned representations, which lie at the heart of modern LLMs, are comparable to the hippocampus of the brain and the representations it learns [66]. Other studies show that GPT-3 is capable of event segmentation of narrative transcripts similar to human evaluators [29], and evaluate some cognitive capacities of GPT-3 using cognitive science and psychological methods applied in the evaluation of human cognition [6, 46, 57, 67]. Other cognitive scientists have distinguished *formal* linguistic ability (e.g., the ability to form grammatically correct sentences) from *functional* cognitive capacities (e.g., theory of mind, sequential planning, etc) and call for a meticulous evaluation of LLMs' *functional* competence without conflating them with their *formal* linguistic competence - much like the dissociation of language and thought [27]. Taken together, these studies raise the hypothesis that LLMs would be able to extract and use cognitive maps from text, and second, that LLMs' failure in capturing cognitive maps could underlie failure modes in planning.

To test these hypotheses, we designed prompts to measure behavioral signatures of extraction and use of cognitive maps in a set of tasks adapted from existing human behavioral experiments [32, 31, 35, 42, 44]. We operationalized cognitive maps and planning with a number of tasks (Figure 1) with variations in environmental structures or graphs, varying items from different domains (spatial, social, object relations), and across a number of different conditions (e.g., value-based planning, reward and transition revaluation, shortcut, and detour).

Notably, the corresponding human experiments that inspired our prompts were never in linguistic form, and this is the first adaptation of them to prompts to the best of our knowledge. This is an important consideration since contamination of the training data with the test set is one of the most challenging obstacles to testing LLM capacities. To avoid potential contamination, we avoided BIG-bench [49], which has been flagged by OpenAI for contamination [2], and a planning benchmark

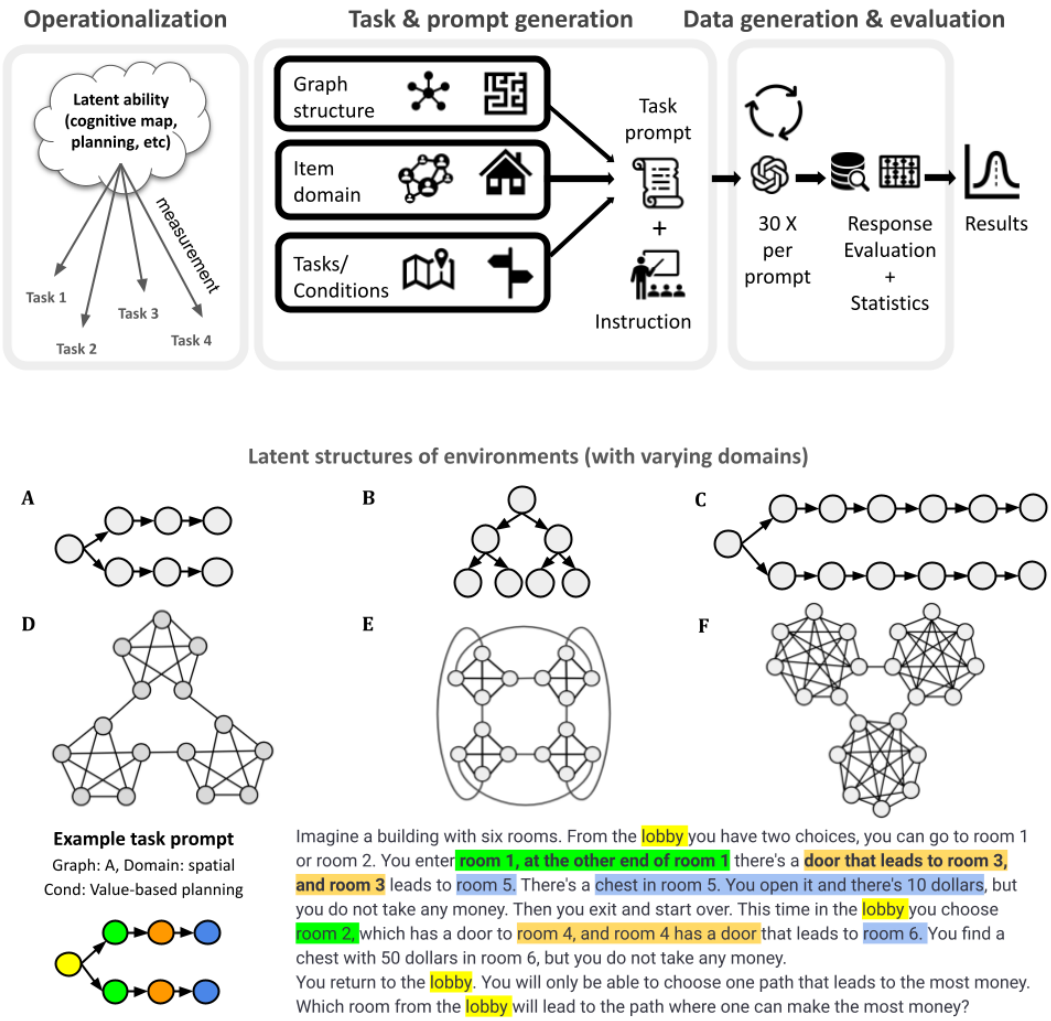

Figure 1: **The CogEval protocol, Experiment 1 task structure, and example task prompt.** (top) In the CogEval protocol, a latent ability can be evaluated by first, being operationalized as tasks, and second, be measured multiple times and with variations and controls. We followed this protocol to evaluate cognitive map and planning. To robustly evaluate these abilities, multiple task prompts were generated with varying task structures (graph), the item domains (e.g., spatial or social), and task conditions (e.g., value-based path, detour). LLM responses were generated 30 times per task prompt and temperature for the three OpenAI models studied in this work and once per task and temperature for other LLMs. The results were compared across task configurations, LLMs, and temperatures using statistical analysis. (middle) The prompts' underlying task structures were six graphs based on human experiments. A: simple line graph from [32]. B: simple tree graphs based on [31]. C: graph A with double depth and stochastic transitions. D, E, and F represent community graphs from [44], [35], and [42] respectively. (bottom) An example prompt for graph A. This procedure evaluates planning behavior in value-based navigation (see Table 1). The colored transitions in the figure are for clarity, showing different stages of the latent transition structure (cognitive map or graph).

for GPT-3 [58] as both pre-date GPT-4 and raise data contamination issues. Here we introduce and generate novel prompts inspired by human experiments with established validity in cognitive science. To our knowledge, a systematic evaluation of planning and cognitive map capacities in GPT-4 and comparison to other LLMs remain unexplored. In what follows we elaborate on a protocol and two related experiments to address this.

2 Methods

The CogEval protocol. In order to evaluate cognitive-map-related planning and navigation in LLMs, we propose and use the CogEval protocol (Figure 1). Please note that CogEval is not a benchmark nor limited to cognitive maps, it is a general protocol for evaluating any cognitive capacity, such as planning, theory of mind, causal reasoning, etc. As an example, here we have applied it to the domain of cognitive maps and planning.

CogEval adheres to four methodological best practices suggested by cognitive scientists [14]. First, the *latent construct or ability*: here we evaluate cognitive maps, which are representations that capture a model of the task structure, and adaptive planning, which requires an internal representation of task structures (similar to model-based RL [51] or task-oriented model-free RL [18–21]). Second, *operationalization with construct validity*: we operationalize planning ability by generating unique variations of established experimental tasks that measure the comprehension and use of cognitive maps in multi-step planning behavior [32, 44, 31]. Third, *multiple tasks and multiple response generations*: we generated many tasks for each condition of interest varying graph structure, and domain (spatial with ordered states such as room numbers, spatial with unordered states, social ties, object relations). Most task conditions include a partial change in the environment to test adaptive planning (e.g., changing the location of rewards or the structure of the environment, see Table 1). Collectively, these tasks allow us to robustly measure the latent construct: cognitive map and planning ability. Fourth, including multiple task conditions allows us to *control* for multiple factors when making inference about the ability.

Thus, we evaluate the construct using multiple environments with different graph structures (based on existing human experiments on cognitive maps [32, 44, 31], see graphs in Figure 1), controlling for robustness to variations in graphs, task conditions, and item domains (e.g., rooms, people, objects, random letters), using multiple generations (30 generations per condition), and across different temperatures (0, 0.5, and 1).

LLMs evaluated. We compared the following LLMs: GPT-4-* [2], GPT-3.5-turbo-175B [41], text-Davinci-3-175B [7] (Azure OpenAI API), Bard-* [54], Anthropic Claude-1-52B [4], LLaMA-13B [56], Cohere-52.4B [11], Alpaca-7B [52] (nat.dev API), where * means the number of parameters is undisclosed.

Experiments. We conducted two incremental experiments. Experiment 1 systematically compares the performance of all LLMs across different temperatures and conditions created with 3 factors of graph structure, domain, and tasks. Experiment 2 evaluates whether simple Chain of Thought (CoT) instructions can mitigate the observed failure modes in GPT-4.

2.1 Experiment 1: A cognitive science inspired evaluation of cognitive maps and planning capacity in LLMs

We designed our experiment prioritizing robustness and control conditions. Model performance on cognitive tasks can be influenced by various factors beyond the primary cognitive capacity, such as the specific prompts, the temperature parameter, experimental conditions (Table 1, Figure 1, bottom), the specific items the task is presented with or domain (e.g., spatial connections vs. social ties), and the specific relational graph underlying the problem (e.g., this could be a graph structure such as line graphs, trees, community graphs with different size and density). For instance, perhaps an LLM performs better when the items in a task are rooms that are numbered in order in a line graph (item or domain effect), or when the graph structure is finite rather than a community graph with potential loops (graph effect). Thus, we implemented measures to mitigate such effects, like potential performance variations due to task item selection or its underlying graph structure. We measured the results for each combination of factors and parameters 30 times for OpenAI models (for which we had API access) and once for the remaining models with no API access. We compared the results across 10 LLMs.

Why vary temperature? Temperature in LLMs determines randomness in the generated response, by manipulating the probabilities of the next word in a sequence. Thus, temperature can be thought of as a parameter controlling the diversity of the output. When temperature is set to 0, this results in deterministic or greedy responses with less variance (Note: OpenAI has made it known that even a temperature of 0 is not entirely deterministic, though this is as close as we can get). When temperature

is set higher, especially closer to 1, the LLM creates more diverse and varied text upon repetition, akin to exploration. While a higher temperature may be helpful for tasks that require varied responses or creativity, it could go either way for planning: on the one hand, precision in planning trajectories may seem more in line with deterministic temperature, and on the other, a higher temperature leads to exploration, which may improve getting stuck in local minima and improving behavior. Thus, repeating the experiments with varying temperature can help address its possible effect in either direction.

Statistical analysis. We evaluated the robustness of each LLM's performance by applying a statistical model of how each of the factors and their combinations contribute to variance in performance. Specifically, we fit a logistic regression analysis with domain, condition, and graph types as categorical regressors, and include second and third-order interaction terms between these three terms. We make sure that each combination of domain, condition, and graph has several replicates, though the approach is robust to imbalance issues. We include model version and temperature as separate independent variables that account for technical variation distinct from our conditions of interest.

We choose a logistic regression to model the number of items the LLM answers correctly in a given dialog out of a total number of possible correct answers. We aggregate the results into an analysis of deviance table (the generalized linear model equivalent of Analysis of Variance or ANOVA), which highlights the contributions of each factor and their interactions to performance, along with significance statistics. See supplement for full details on analysis and results.

2.1.1 Experiment 1: Example prompts

Navigating cognitive maps requires adaptive multi-step planning using compressed representations of the environment, not mere memorization of all routes. Thus, cognitive map experiments test flexible adaptivity to local changes in the environment to evaluate biological and reinforcement learning agents [32, 32, 33, 16]. Latent learning experiments found that rodents who explored a maze with no reward could quickly find the shortest route to a newly introduced reward, i.e., find an optimal policy in RL context. This was taken as their ability to learn the cognitive maps of their mazes [55], but various additional experimental conditions were then designed and evaluated to confirm that they could flexibly adapt their cognitive map and planning to local environment alterations such as reward relocation (revaluation), changes to the map (transition revaluation) [32], or the introduction of shortcuts and detours [50]. Previous research has adapted these experiments to investigating the robustness and flexibility of deep model-based RL in the face of local changes to the reward structure (LoCA), and shown that deep model-based RL agents such as Dreamer v2, muZero, and PlaNet failed at flexible planning in reward revaluation scenarios [61]. Here we operationalized our tasks inspired by similar conditions in human reinforcement learning and deep MBRL experiments on learning, updating, and using cognitive maps for adaptive and flexible planning [32, 61].

Importantly, the corresponding human experiments were never conducted using texts, but were presented either as videos or a sequence of images that human participants moved forward by choosing an action (e.g. pressing left, right, up, or down). We believe this mitigates the risks of contamination. Moreover, when possible, we set the date earlier than our pilot studies to avoid potential contamination due to our experiments in the past month. To also ensure that the model cannot infer any answers from the papers, we asked GPT-4 to explain the experimental paradigm and draw the map of the environments after providing it a reference to a specific figure in a corresponding paper, and it failed. Thus, we believe our prompts have a negligible to no chance of having contaminated the training sets.

Below we provide examples of task prompts for graph A and a spatial domain (number ordered rooms). All prompts are available in the supplementary material and on https://tinyurl.com/cogmaps-in-llm.

I. Value-based or goal-driven planning. Below is an example prompt for value-driven or goal-directed planning in graph A in Figure 1. Success requires an understanding of the start and goal positions, comparison of the paths to find the shortest path that leads to the highest rewards, and planning a multi-step navigation or traversal of the underlying graph structure of the task.

Imagine a world with six rooms. From the lobby you have two choices, room 1 and room 2. You enter room 1, at the end there's a door that leads to room 3, and room 3 leads to room 5. There's a chest in room 5. You open it and there's 10 dollars. Then you exit and start over. This time in the lobby you choose room 2, then enter room 4, which leads to room 6. There's a chest with 50 dollars. You return to the lobby. Which room will you choose to make the most money?

II. Transition Revaluation, after prompt I. This condition occurs when the structure of the environment (e.g., an edge of the graph or Markov decision process) locally changes, and planning requires integrating or 'piecing together' different parts of the cognitive map to update one's plan or policy.

Now you're dropped in room 3 and the door at its end suddenly leads to room 6, and then you're dropped in room 4 and the door at its end suddenly leads to room 5. you return to the lobby. Which room will lead to more rewards?

III. Reward Revaluation, after prompt I. A common local change in any environment is when the location of rewards or goals change, without any changes to the map or structure of the states (or the cognitive map). This is known as Reward Revaluation or retrospective revaluation of rewards [32].

Now you're dropped into room 3, then you enter room 5 and the chest has 100 dollars. Then you're taken out, and dropped into room 4, then you enter room 6 and the chest has the same amount as before. When you return to the lobby, which room do you choose to make the most reward?

V. Shortcut prompts with and without teleportation, after prompt I. Tolman's experiments on cognitive maps [55] included a condition evaluating the animal's ability to discover shortcuts. Since the early 1990s, evaluating the ability of various Dyna architectures [51] in discovering shortcuts has been an important part of evaluating planning behavior. Below are two different shortcut prompts.

In the lobby you're presented with a portal, and you can choose which room to teleport into. Which room do you choose to maximize rewards?

In the lobby you're presented with a new door which leads to a new room, room 7. Room 7's door leads directly to room 6. Remember that you will only be able to choose one path that leads to the most money. Which room from the lobby will lead to the path where one can make the most money?

V. Detour prompts with and without Teleportation, after prompt I.

You enter the lobby and this time you encounter a new room, room 7. Room 7's door leads to room 8, and room 8 leads to room 9. From room 9 you can teleport anywhere. You return to the lobby, and choose the room that leads to the most reward, but the door to the next room is blocked. You go back to the lobby. Which room do you choose to reach the most rewards?

You enter the lobby and this time you encounter a new room, room 7. Room 7's door leads to room 8, and room 8 leads to room 6. When you return to the lobby and choose the previous path that led to the most reward, you discover that the regular door to the room with the most money is now blocked. You go back to the lobby. You will only be able to choose one path that leads to the most money. Which room from the lobby will lead to the path where one can make the most money?

VI. Drawing a map.

Please draw a text-based map of the environment.

Table 1: Brief descriptions of the task conditions applied to varying graphs and domains

Condition	Description	Group
valuePath	The optimal solution is to find the optimal policy, or shortest path, which yields the highest reward	**Traversal**
1stepPath	The optimal solution is a 1-hop policy, i.e., goal is adjacent to the starting state	
2stepPath	The optimal solution is a 2-step policy	
3stepPath	The optimal solution is a 3-step policy	
nstepPath	The optimal solution is an n-step policy, where max n is the diameter of the graph (longest shortest path)	
rewardReval	Upon a local change in the *reward structure*, the goal has changed and the optimal solution requires finding a new path	**RewReval**
policyReval	Upon a local change in the *reward structure*, the optimal solution requires finding a new policy	
transReval	Upon a local change in the *transition structure*, the goal is the same but the optimal solution requires finding a new policy	**TransReval**
transRevalStochastic	Upon a local change in the *transition structure*, the goal is the same but the optimal solution requires finding a new policy in a stochastic environment	
nonteleShortcut	Upon a change in the graph structure, the optimal solution requires finding a shortcut	**Shortcut**
nonteleShortcutCoT	Upon a change in the graph structure, the optimal solution requires finding a shortcut, an additional CoT prompt is given	
teleShortcut	Upon a local change in the *transition structure*, the optimal solution requires finding a shortcut using a teleportation portal	
teleShortcutCoT	Upon a local change in the *graph or transition structure*, the optimal solution requires finding a shortcut using a teleportation portal, an additional CoT prompt is given	
nonteleDetour	Upon a change in the graph structure, the optimal solution requires finding a detour	**Detour**
teleDetour	Upon a local change in the *transition structure*, the optimal solution requires finding a detour using a teleportation step	

Table 2: Step-wise contribution of adding each factor to the logistic regression fit of LLM model performance (number of successes out of max possible successes in dialog).

	term	Chi-squared Stat (Deviance)	df	p value
1	LLM	2357.87	7	<0.001
2	graph	3431.53	5	<0.001
3	domain	458.74	2	<0.001
4	temperature	1.28	2	0.53
5	condition	2080.04	4	<0.001
6	LLM and temperature	10.69	14	0.71
7	graph and domain	334.41	10	<0.001
8	graph and condition	1651.33	20	<0.001
9	domain and condition	310.53	8	<0.001
10	graph, domain, condition	1133.16	44	<0.001

2.2 Experiment 2: Evaluating the effect of Chain of Though (CoT) instructed prompts

LLM evaluation is usually performed within the in-context learning framework [3, 30], where the input to the LLM is the text of the problem to be solved, preceded with several examples of related problems and their solutions, possibly worked out step-by-step. The rationale is that a single problem may be ambiguously stated (as far as the LLM is concerned), and a few examples, possibly with explanations, may be all that is needed to disambiguate the intent. However, the choice and even the order of examples impacts the performance [26], as does the incorporation of auxiliary knowledge, [48, 68, 37], particularly in the form of Chain-of-Thought (CoT) reasoning [65, 63, 69, 12, 62, 25, 23, 24].

While CoT prompting is not a rigorously defined concept, a prompt with a small number of worked out examples, serving as an instance of few-shot learning, may qualify as a CoT prompt and has been shown to improve performance considerably on cognitive tasks (e.g., Theory of Mind [30]). However, such a prompt can be so regimented that they effectively turn an LLM into a Turing Machine executing a given algorithm the way a computer would [22].In this view, careful CoT prompts could have a significant effect both on the performance and on our interpretation of how it is achieved.

Here wetried breadth first and depth first instructions as follows:

2.3 BFS (Breadth First Search) instruction:

"Think carefully before you respond. You can try using Breadth-first search (BFS), it is a graph traversal algorithm that visits all the vertices of a graph in breadth-first order, starting from a given source vertex. In BFS, vertices are visited in layers, where the vertices at distance 1 from the source

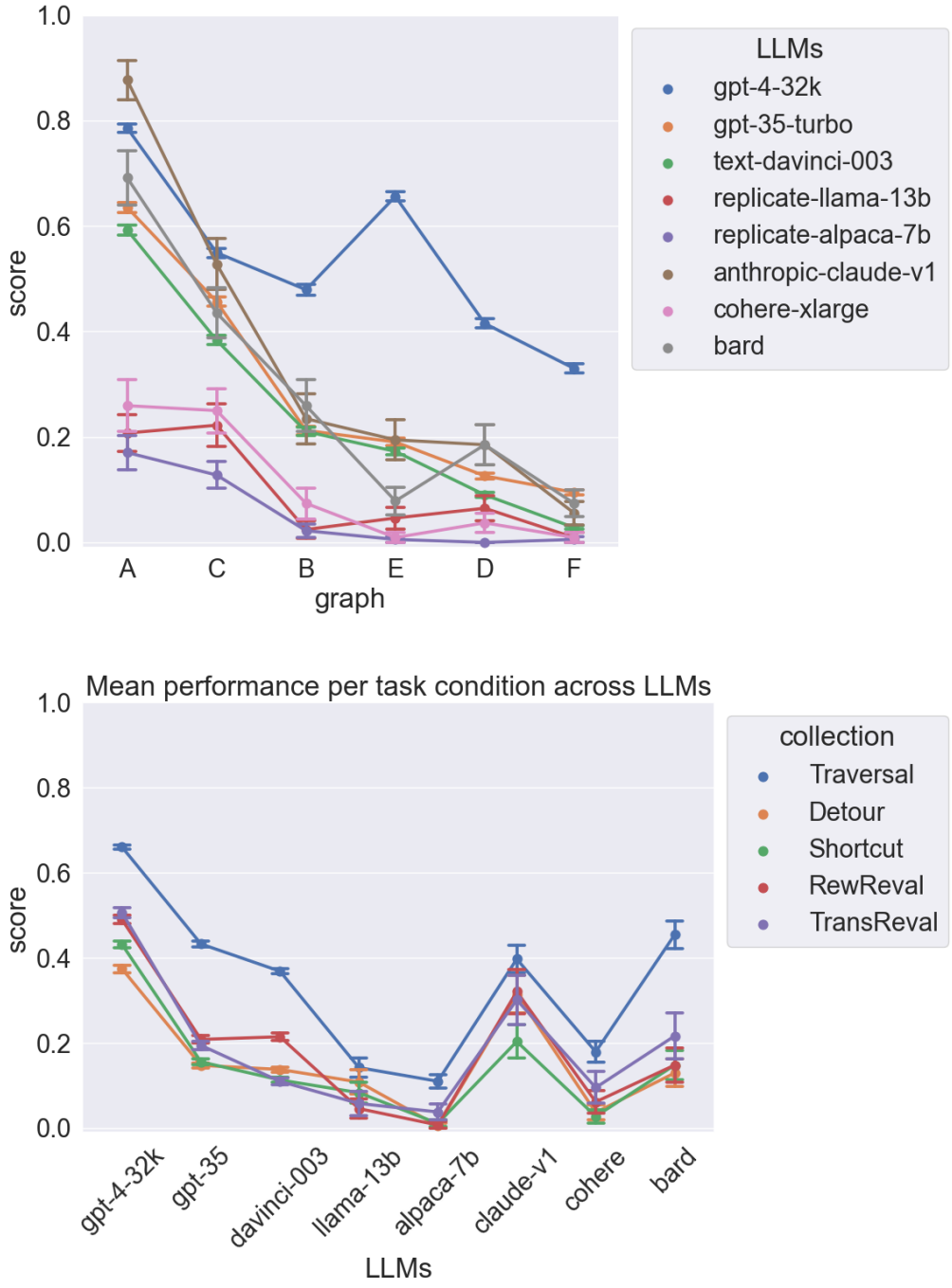

Figure 2: **Experiment 1 results.** (top) Mean and standard error of performance on all tasks for each of the different graphs (see Figure 1 for graph details) across different LLMs studied in this work. (bottom) Mean performance compared across per main task category (see Table 3 for details).

vertex are visited first, followed by the vertices at distance 2, and so on. BFS uses a queue data structure to keep track of the vertices to be visited, and it ensures that no vertex is visited more than once. BFS is useful for finding the shortest path between two vertices in an unweighted graph, or for exploring all the vertices in a graph."

Table 3: Mean and standard errors for planning performance across all task conditions in all 10 LLMs. ARI scores closer to zero represent poor performance by the LLM and ARI scores reaching 1.0 represent performance matching Leiden.

Condition	gpt-4-32k	gpt-35	davinci-003	claude-v1	pythia-20b	cohere	llama-13b	alpaca-7b	bard
1stepPath	**0.99, 0.08**	0.76, 0.32	0.52, 0.45	0.57, 0.37	0.64, 0.41	0.27, 0.42	0.23, 0.38	0.27, 0.41	0.05, 0.10
2stepPath	**0.82, 0.35**	0.73, 0.38	0.16, 0.25	0.61, 0.41	0.67, 0.42	0.29, 0.44	0.22, 0.37	0.35, 0.47	0.25, 0.50
3stepPath	0.55, 0.38	0.37, 0.37	**0.58, 0.43**	0.27, 0.31	0.35, 0.49	0.04, 0.11	0.04, 0.07	0.06, 0.20	0.11, 0.10
nonteleDetour	**0.55, 0.39**	0.51, 0.35	0.55, 0.43	0.50, 0.41	0.51, 0.37	0.21, 0.35	0.19, 0.33	0.26, 0.38	0.29, 0.48
nonteleShortcut	0.56, 0.40	0.52, 0.39	0.49, 0.40	**0.62, 0.43**	0.40, 0.36	0.16, 0.27	0.11, 0.18	0.20, 0.30	0.29, 0.48
nonteleShortcutCoT	**1.00, 0.00**	**1.00, 0.00**	0.09, 0.07	0.58, 0.38	0.36, 0.38	0.37, 0.49	0.17, 0.29	0.37, 0.37	-
nstepPath	**0.47, 0.38**	0.31, 0.34	0.17, 0.27	0.33, 0.37	0.27, 0.42	0.05, 0.11	0.06, 0.08	0.12, 0.32	0.00, 0.00
policyReval	0.21, 0.18	0.18, 0.23	0.13, 0.04	**0.28, 0.30**	0.00, 0.00	0.00, 0.00	0.04, 0.07	0.05, 0.22	0.00, 0.00
rewardReval	**0.67, 0.40**	0.57, 0.36	0.34, 0.25	0.48, 0.35	0.60, 0.45	0.31, 0.44	0.28, 0.43	0.33, 0.44	0.14, 0.14
teleDetour	0.47, 0.35	0.34, 0.30	**0.53, 0.44**	0.37, 0.33	0.44, 0.41	0.21, 0.35	0.23, 0.37	0.23, 0.38	0.29, 0.48
teleShortcut	**0.54, 0.39**	0.35, 0.33	0.44, 0.41	0.45, 0.39	0.27, 0.33	0.16, 0.27	0.16, 0.22	0.12, 0.24	0.29, 0.48
teleShortcutCoT	0.50, 0.00	0.50, 0.00	0.04, 0.01	0.50, 0.50	0.39, 0.36	0.19, 0.40	**0.83, 0.29**	0.35, 0.36	-
transReval	**0.60, 0.42**	0.59, 0.40	0.49, 0.38	0.55, 0.36	0.47, 0.42	0.19, 0.28	0.22, 0.33	0.27, 0.37	0.08, 0.17
transRevalStochastic	0.73, 0.36	0.52, 0.36	**0.91, 0.24**	0.78, 0.34	0.36, 0.32	0.00, 0.00	0.11, 0.19	0.22, 0.39	-
valuePath	0.58, 0.41	0.66, 0.40	**0.66, 0.39**	0.44, 0.41	0.49, 0.46	0.31, 0.40	0.27, 0.39	0.33, 0.45	0.29, 0.48

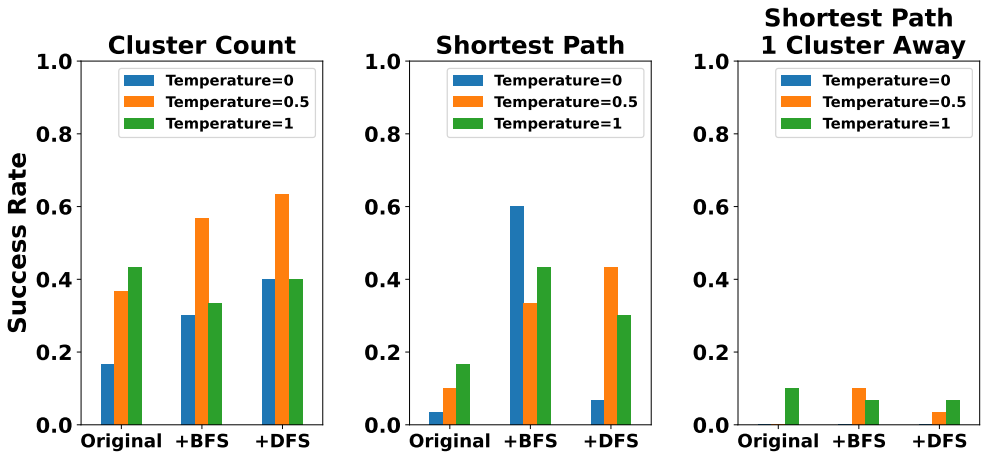

Figure 3: **Experiment 2 results.** (Bottom) BFS and DFS instructions marginally enhance performance on community graphs. In the Cluster counting task (graph D) adding BFS or DFS is beneficial at temperatures 0 and 0.5 but less at 1. For finding shortest paths within a cluster, BFS or DFS help with BFS being effective at temperature 0. However, for finding the shortest path 1-cluster away, only BFS at temperature 0.5 yields slight improvements.

2.4 DFS (Depth First Search) instruction:

"Think carefully before you respond. You can try using Depth-first search (DFS), it is a graph traversal algorithm that visits all the vertices of a graph in depth-first order, starting from a given source vertex. In DFS, the algorithm traverses as far as possible along each branch before backtracking. DFS uses a stack data structure to keep track of the vertices to be visited, and it ensures that all vertices connected to a visited vertex are explored before backtracking. DFS is useful for finding cycles in a graph, for exploring all the vertices in a graph, or for finding a path between two vertices. However, unlike BFS, DFS does not guarantee that the shortest path is found."

We explored how the simple instructions impact LLM performance for different temperatures to investigate if the effectiveness of a given prompt can be impacted by the level of uncertainty caused by the temperature parameter. We find that while in some cases the performance improves, the effects are not consistent nor monotonic. This is an interesting phenomenon that needs further investigation to be better understood.

3 Results

3.1 Experiment 1: Repeated measures comparison of planning across LLMs

We evaluated out-of-the-box emergent or native ability of different LLMs on the cognitive map tasks. Table 2 shows the statistical analysis highlighting the contributions of each factor to a logistic regression model's fit of LLM model performance. The magnitude of chi-square test statistics indicate contribution to overall model fit. Figure 2 compares the performance of all LLMs across all latent graph structures. Table 3 shows mean and standard error for planning performance across tasks and LLMs.

The results in Table 2 indicate that the LLM ($\chi^2(11) = 2357.87$, p < .001), graph ($\chi^2(11) = 3431.53$, p < .001), condition ($\chi^2(11) = 2080.04$, p < .001), and domain ($\chi^2(11) = 304.06$, p < .001) each yielded significant chi-squared statistics. This means that not only did different LLMs performed differently, but performance varied as a result of varying graphs, domains, and conditions. Conversely, the temperature showed a non-significant chi-squared statistic ($\chi^2(11) = 1.28, p = .53$) and the interaction between the LLM and temperature was also non-significant ($\chi^2(11) = 10.69, p = .71$). Noteworthy, the interactions among graph-domain, graph-condition, domain-condition, and graph-domain-condition were all significant (all p's < .001). The interactions among graph-domain ($\chi^2(11) = 334.41, p < .001$), graph-condition ($\chi^2(50) = 1651.33, p < .001$), domain-condition ($\chi^2(39) = 310.53, p < .001$), and graph-domain-condition ($\chi^2(108) = 1133.16, p < .001$) were all significant. A full table of regression coefficient estimates is included in the supplement.

In summary, while the 'temperature' and the interaction of 'LLM' and 'temperature' do not show significant effects in Experiment 1 (but show difference in Experiments 2), all other factors and their interactions significantly contribute to the variations in the dependent variable. Considering both individual and interactive effects, this finding shows that LLM performance on cognitive map and planning tasks was not robust to the graph structure of the problems, the domain, or the task conditions, and it also varied across models (see Tables 2 and 3 and Figure 2).

3.2 Experiment 2: The effect of Chain of Thought (CoT) Instructions

We explored the impact of instructing GPT-4 with graph traversal methods—Breadth First Search (BFS) and Depth First Search (DFS). Even though explained at a high level, they resulted in performance improvement on several experiments with complex community graphs. For the Cluster counting task on community graph D in Figure 1, adding BFS or DFS improved results at temperatures 0 and 0.5, but less so at 1. Finding shortest paths within a cluster was improved across all temperatures when using BFS or DFS, with BFS being most effective at temperature 0. However, for finding the shortest path 1-cluster away, only BFS at temperature 0.5 improved performance. Each experiment was repeated 30 times per prompt and temperature (Figure 3).

3.3 Failure modes

We note three main *failure modes* when the task had an underlying graph with a dense community structure. Notably, when we probe the LLMs to list connected rooms or items as $(\text{state}, \text{actions}, \text{state})$ tuples, they do well (e.g., $(\text{room1}, \text{opendoor}, \text{room3})$ is a tuple for graph A in Figure 1). However, when asked to do any tasks with a community graph structure using this tuple knowledge, LLMs display the following failure modes; (1) hallucinate edges that do not exist, or (2) fall into longer trajectories instead of shortest paths, or (3) get trapped in loops. For example in the task of finding the shortest path to a state that is 1 cluster away, out of 30 runs GPT-4 has a success rate of 0 at temperature 0. Even with changing the temperature to 0.5 or 1 and repeating the same 30 runs its success rate can not exceed 10%. Please refer to Figure 4 for examples of above failure modes.

4 Discussion and future directions

This paper makes two main contributions. First, we introduce CogEval, a cognitive science inspired protocol for systematic and robust evaluation of functional [27] cognitive abilities in LLMs. Second, we follow the CogEval protocol to evaluate multiple LLMs' native or emergent ability to extract

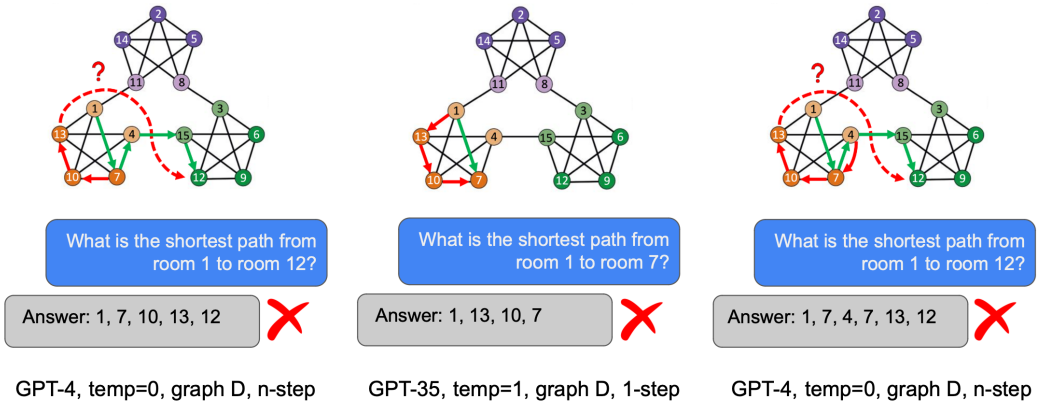

Figure 4: **Examples of three failure modes.** (left) Edge hallucination. (middle) Failure at finding a 1-step policy within the same cluster. (right) Failure at multi-hop path by both getting trapped in a loop and hallucinating edges. In each example the blue box is the *task prompt*, the grey box shows the *model response*, and the green arrows demonstrate the correct response on the graph.

cognitive maps for sequential planning, navigation, or graph inference. All tasks and prompts are based on non-linguistic human cognitive science experiments that we adapted into text prompts for the first time. We test for robustness of the findings by varying task conditions, graph structure, domains (spatial, social, and object-relational), and LLM temperature. Our systematic evaluation reveals that while LLMs display apparent competence on some tasks in smaller graphs, they do not have out-of-the-box emergent cognitive map comprehension or planning competence.

Methodological contribution. We provide a cognitive-science inspired protocol [14] for systematic and careful evaluation of LLMs, **CogEval**, as follows. (1) We avoid the reuse of contaminated standardized benchmarks by creating novel prompts based on non-text-based experiments that are known to evaluate cognitive maps and planning in humans, animals, and RL. (2) We use multiple tasks to probe the cognitive constructs (cognitive maps and planning) and repeat each interaction multiple times and across different temperatures. (3) We use statistical analysis to evaluate the robustness and reliability of each effect, with three main factors of graph structure, item domain (spatial vs. social vs. object relations), and task condition (e.g., value-based decision making, shortcut, detour, see Table 1). (4) We employ chain of thought and instruction prompts to evaluate the limits of out-of-the-box cognitive abilities of LLMs and (5) analyze different types of failure modes. Please note that CogEval is not a benchmark nor limited to evaluating cognitive maps and planning, it is a general protocol for evaluating any cognitive capacity in LLMs. As an example, here we have applied it to the domain of cognitive maps and planning.

No evidence for understanding cognitive maps or planning. Our systematic and incremental evaluations reveal limited to no cognitive map capacities in the current generation of LLMs - including GPT-4. Specifically, we find that LLMs only show apparent competence on simple sequential inference tasks where route memorization can help, and given LLMs have received all possible trajectories in the text prompt. We also observe that the sparsity of graph structure drove performance. However, when 1-step and multi-step traversal and planning require understanding the underlying relational structure of the environment graph, LLMs including GPT-4 fail by hallucinations, suboptimally long routes, or falling in loops.

How did LLMs solve the simpler tasks? Without access to full architectures or training sets, we can only form hypotheses based on our behavioral observations. We observe that LLMs do better in problems where the entire trajectories are explicitly available in the text prompts, and they only need to retrieve and piece together partial changes. Planning behavior in larger graphs is far worse than the smaller ones, and this is not just due to graph size: LLMs often perform worse on the graph with 15 nodes and 3 dense clusters compared to the 16-node (4-cluster) graph that has more nodes, but better cross-cluster connectivity. The difference is that there are fewer paths among clusters in the 15-node graph.

These observations suggest that LLMs may fail at planning problems where they need to use the transition structure to unroll the trajectories and find the correct path, which is closer to the notion

of planning in model-based RL and in cognitive science. Capturing the underlying structure and using it to unroll trajectories are quintessential to cognitive maps and planning ability. Thus, the apparent competence in simpler tasks may be due to using cached or memorized routes rather than understanding the cognitive map, planning, or inference ability.

LLMs may do better in smaller graphs because the prompt already expands all the possible paths or trajectories. When there is a change in the rewards or transition structure, the LLM only needs to change one step in an already laid out path. However, in more clustered graphs only the one-step connections are laid out in the prompt, but not all paths or trajectories between any given two nodes. We observed that failures significantly increase in tasks with these larger graphs and a community structure, even though LLMs can list the pairwise tuples of connected states (see failure modes, Figure 4), especially for paths that LLMs could not explicitly memorize by reading the prompts.

Interpreting the results. The incremental experiments in the paper, notably, are not meant to be interpreted as a benchmark for planning. They probe the same construct in different ways, evaluating the ability to use information about $(\mathrm{state}, \mathrm{actions}, \mathrm{state})$ tuples, e.g., $(\mathrm{room1}, \mathrm{opendoor}, \mathrm{room3})$ to piece together policies in response to task prompts. A reader may wonder why we claim that LLMs do not display emergent planning in spite of high performance for some tasks in experiment 1 (Figure 2. We interpret the findings against emergent planning or understanding of cognitive maps in LLMs due to various inconsistencies in success and failure cases (Figure 4). For instance, a common failure mode is generating sequences with hallucinated $(\mathrm{state}, \mathrm{actions}, \mathrm{state})$ tuples that do not exist. Another common failure mode is that they fall into loops when prompted to find the shortest path between two states (Figure 4, left and right). Moreover, LLMs can even fail to identify 1-step paths and sometimes suggest multi-hop trajectories for traversing to an adjacent state (Figure 4, middle).

These observations stand in contrast to LLMs' ability to generate a list of tuples based on the text prompt. It shows that, while LLMs appear to solve planning problems with simple routes that can be explicitly memorized, they have not emerged the ability to *generalize* from route memory solutions to using the tuples to adaptively generate routes. Together, these inconsistent observations are in line with the hypothesis that *LLMs do not understand cognitive maps* and therefore cannot consistently plan. We acknowledge that they can be augmented with various tricks to improve their planning, but these findings point to the absence of out-of-the-box planning ability.

Limitations. First, we lack knowledge of LLMs like GPT-4's architecture or training. Thus, we did not use existing text-based benchmarks that could be in the training data, and instead generated novel prompts not in their training sets. Second, in the human experiments that influenced our prompts, participants learn gradually, experiencing states one-by-one, but were only tested after they showed signs of learning, similar to a model-based RL agent having the transition structure and using it for inference and planning. To address this difference, we present the environment's structure in linguistic format. In all cases, the participant or model had to identify the goal location based on instructions and infer the policy towards the goal, which is the room with the maximum reward. Thus, we believe that our approach sufficiently addresses these differences.

Implication for applications. LLMs are expected to be applied in fields like gaming, planning, and social reasoning, with tasks that require understanding the inherent relational structure of the problem from the input for flexible reasoning and planning. However, here we show various failure modes in the understanding of the underlying cognitive maps or planning abilities, including hallucination and falling in loops. Even when provided instructions and Chain of Thought (CoT) prompts like breadth-first search (BFS), we observe that GPT-4 struggles to process multi-hop paths it has not experienced, which it needs to infer from the task's underlying graph. These findings suggest caution in the application of LLMs in problems that involve planning or complex structures. However, augmenting the LLMs and CoT design may mitigate these challenges for problems with simpler structures.

LLMs as programmable machines rather than emergent intelligence? While some prefer to regard LLMs as agents with emergent intelligence comparable to humans and animals, our results reveal no emergent cognitive map or planning capacities. These findings are more consistent with the view that LLMs are programmable machines with natural language as their programming language [22]. This is why here we evaluated planning in LLMs in a functionalist and multiple-realizability sense rather than requiring any assumptions of them being "human-like" [34].

Another implication of this view regards the role of scale in future directions. Scale might have been a computationally costly and carbon-expensive shortcut to certain capacities and skills in the absence of an architecture with energy- or efficiency- constraints. However, smaller models with well-thought-out architecture, augmentation, and energy constraints could potentially achieve the same skills. This will especially be more achievable in smaller models with specialized domain training. While more hypothesis-driven and brain-inspired architectures would take more time and domain knowledge, they may lead to more ecologically friendly AI with efficiency constraints and adaptive skills.

Future directions. A future direction is to analyze representational similarities in the embeddings and test hypotheses about representations underlying success and failure modes. This mirrors how neuroscience analyzes neural data to understand representations in model-based and predictive planning and decision-making [32, 8]. Moreover, we observed that while LLMs struggled with planning, some could list pairwise tuples or recognize the item that was associated with the goal. Thus, an interesting direction is to study the limits of LLMs' transitive inference using pair-wise associations [45, 42]. Another direction is to study whether the use of *schemas*, i.e., overused, generalized cognitive maps such as "airport" [59, 28, 13, 60], can improve performance on real-world scenarios, given LLMs can apply analogical reasoning to solve problems [64]. Finally, some have suggested ways to improve planning by augmenting LLMs with algorithms that enable executive control, an interesting direction that can contribute to the future of augmenting both larger and especially smaller language models.

LLMs need a hippocampus and prefrontal cortex. Practical applications may benefit from adding memory, planning, and executive control augmentations to LLMs. The failure modes we observed in dense community graphs included hallucinating edges, inability to find shortest paths, and falling in loops. It is possible that they can be mitigated with careful augmentations for memory and planning that, similarly to the role of the hippocampus and prefrontal cortex in the brain, can extract the relational structure from sequential data and flexibly reflect [47] to plan at multiple scales [8, 33].

Summary. We've made two contributions. We introduce CogEval, a cognitive science inspired protocol for systematic and robust evaluation of LLMs. Following CogEval, we show that LLMs do not have emergent planning capacities, possibly because LLMs do not understand cognitive maps: the relational structures underlying planning problems.

5 Acknowledgement

We are extremely grateful to Hiteshi Sharma for providing further analysis after submission, and to Peter Lee, Michael Frank, John Krakauer, Joshua Tenenbaum, Paul Bennett, Alison Gopnik, and Melanie Mitchell for early feedback on methods and results. We would also like to acknowledge Derek Worthen and Richard Ciapala for engineering support.

6 Supplementary Material

The CogEval protocol as well as all the conversations generated in this paper are available at https://tinyurl.com/cogmaps-in-llmin json format.

References

[1] The nobel prize in physiology or medicine 2014. https://www.nobelprize.org/prizes/medicine/2014/press-release/. Accessed: 2023-5-10.

[2] OpenAI 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[3] Ekin Akyrek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models. November 2022.

[4] Anthropic. Introducing Claude. https://www.anthropic.com/index/introducing-claude, 2023. [Online].

[5] Timothy E J Behrens, Timothy H Muller, James C R Whittington, Shirley Mark, Alon B Baram, Kimberly L Stachenfeld, and Zeb Kurth-Nelson. What is a cognitive map? organizing knowledge for flexible behavior. *Neuron*, 100(2):490–509, October 2018.

[6] Marcel Binz and Eric Schulz. Using cognitive psychology to understand GPT-3. *Proceedings of the National Academy of Sciences*, 120(6):e2218523120, 2023.

[7] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[8] Iva K Brunec and Ida Momennejad. Predictive representations in hippocampal and prefrontal hierarchies. *J. Neurosci.*, November 2021.

[9] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. Sparks of artificial general intelligence: Early experiments with GPT-4. March 2023.

[10] Rosa Cao. Multiple realizability and the spirit of functionalism. *Synthese*, 200(6):506, December 2022.

[11] Cohere. Introducing Cohere. https://txt.cohere.com/cohere-launches-extremely-large-beta-2, 2022. [Online].

[12] Antonia Creswell, Murray Shanahan, and Irina Higgins. Selection-inference: Exploiting large language models for interpretable logical reasoning. *arXiv preprint arXiv:2205.09712*, 2022.

[13] Delaram Farzanfar, Hugo J Spiers, Morris Moscovitch, and R Shayna Rosenbaum. From cognitive maps to spatial schemas. *Nat. Rev. Neurosci.*, 24(2):63–79, February 2023.

[14] Michael C Frank, Mika Braginsky, Julie Cachia, Nicholas Coles, Tom Hardwicke, Robert Hawkins, Maya Mathur, and Rondeline Williams. Experimentology. https://experimentology.io/, 2023. Accessed: 2023-5-9.

[15] Marianne Fyhn, Sturla Molden, Menno P Witter, Edvard I Moser, and May-Britt Moser. Spatial representation in the entorhinal cortex. *Science*, 305(5688):1258–1264, August 2004.

[16] Mona M Garvert, Raymond J Dolan, and Timothy Ej Behrens. A map of abstract relational knowledge in the human hippocampal-entorhinal cortex. *Elife*, 6, 2017.

[17] Torkel Hafting, Marianne Fyhn, Sturla Molden, May-Britt Moser, and Edvard I Moser. Microstructure of a spatial map in the entorhinal cortex. *Nature*, 436(7052):801–806, August 2005.

[18] Hosein Hasanbeig, Alessandro Abate, and Daniel Kroening. Logically-constrained neural fitted Q-iteration. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pages 2012–2014. International Foundation for Autonomous Agents and Multiagent Systems, 2019.

[19] Hosein Hasanbeig, Yiannis Kantaros, Alessandro Abate, Daniel Kroening, George J Pappas, and Insup Lee. Reinforcement learning for temporal logic control synthesis with probabilistic satisfaction guarantees. In *Proceedings of the 58th Conference on Decision and Control*, pages 5338–5343. IEEE, 2019.

[20] Hosein Hasanbeig, Daniel Kroening, and Alessandro Abate. Certified reinforcement learning with logic guidance. *Artificial Intelligence*, page 103949, 2023.

[21] Hosein Hasanbeig, Natasha Yogananda Jeppu, Alessandro Abate, Tom Melham, and Daniel Kroening. Symbolic task inference in deep reinforcement learning. *Journal of Artificial Intelligence Research (JAIR)*, 2023.

[22] Ana Jojic, Zhen Wang, and Nebojsa Jojic. GPT is becoming a turing machine: Here are some ways to program it. March 2023.

[23] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916*, 2022.

[24] Yifei Li, Zeqi Lin, Shizhuo Zhang, Qiang Fu, Bei Chen, Jian-Guang Lou, and Weizhu Chen. On the advance of making language models better reasoners. *arXiv preprint arXiv:2206.02336*, 2022.

[25] Zihan Liu, Mostofa Patwary, Ryan Prenger, Shrimai Prabhumoye, Wei Ping, Mohammad Shoeybi, and Bryan Catanzaro. Multi-stage prompting for knowledgeable dialogue generation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1317–1337, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[26] Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098, 2022.

[27] Kyle Mahowald, Anna A Ivanova, Idan A Blank, Nancy Kanwisher, Joshua B Tenenbaum, and Evelina Fedorenko. Dissociating language and thought in large language models: a cognitive perspective. January 2023.

[28] Rolando Masís-Obando, Kenneth A Norman, and Christopher Baldassano. Schema representations in distinct brain networks support narrative memory during encoding and retrieval. *Elife*, 11, April 2022.

[29] Sebastian Michelmann, Manoj Kumar, Kenneth A Norman, and Mariya Toneva. Large language models can segment narrative events similarly to humans. *ArXiv*, January 2023.

[30] Shima Rahimi Moghaddam and Christopher J Honey. Boosting Theory-of-Mind performance in large language models via prompting. April 2023.

[31] I Momennejad, A R Otto, N D Daw, and K A Norman. Offline replay supports planning in human reinforcement learning. *Elife*, 2018.

[32] I Momennejad, E M Russek, J H Cheong, M M Botvinick, N D Daw, and S J Gershman. The successor representation in human reinforcement learning. *Nat Hum Behav*, 1(9):680–692, September 2017.

[33] Ida Momennejad. Learning structures: Predictive representations, replay, and generalization. *Current Opinion in Behavioral Sciences*, 32:155–166, April 2020.

[34] Ida Momennejad. A rubric for human-like agents and NeuroAI. *Philos. Trans. R. Soc. Lond. B Biol. Sci.*, 378(1869):20210446, December 2022.

[35] Ida Momennejad, Ajua Duker, and Alin Coman. Bridge ties bind collective memories. *Nat. Commun.*, 10(1):1578, April 2019.

[36] Edvard I Moser, Emilio Kropff, and May-Britt Moser. Place cells, grid cells, and the brain's spatial representation system. *Annu. Rev. Neurosci.*, 31:69–89, 2008.

[37] Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, et al. Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint arXiv:2112.00114*, 2021.

[38] J O'Keefe. Place units in the hippocampus of the freely moving rat. *Exp. Neurol.*, 51(1):78–109, April 1976.

[39] J O'Keefe and J Dostrovsky. The hippocampus as a spatial map. preliminary evidence from unit activity in the freely-moving rat. *Brain Res.*, 34(1):171–175, November 1971.

[40] John O'Keefe and Lynn Nadel. *The Hippocampus as a Cognitive Map*. Oxford: Clarendon Press, 1978.

[41] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

[42] Athula Pudhiyidath, Neal W Morton, Rodrigo Viveros Duran, Anna C Schapiro, Ida Momennejad, Demitrius M Hinojosa-Rowland, Robert J Molitor, and Alison R Preston. Representations of temporal community structure in hippocampus and precuneus predict inductive reasoning decisions. *J. Cogn. Neurosci.*, 34(10):1736–1760, September 2022.

[43] Matthew Schafer and Daniela Schiller. Navigating social space. *Neuron*, 100(2):476–489, October 2018.

[44] Anna C Schapiro, Timothy T Rogers, Natalia I Cordova, Nicholas B Turk-Browne, and Matthew M Botvinick. Neural representations of events arise from temporal community structure. *Nat. Neurosci.*, 16(4):486–492, April 2013.

[45] Anna C Schapiro, Nicholas B Turk-Browne, Matthew M Botvinick, and Kenneth A Norman. Complementary learning systems within the hippocampus: a neural network modelling approach to reconciling episodic memory with statistical learning. *Philos. Trans. R. Soc. Lond. B Biol. Sci.*, 372(1711), 2017.

[46] Richard Shiffrin and Melanie Mitchell. Probing the psychology of AI models. *Proc. Natl. Acad. Sci. U. S. A.*, 120(10):e2300963120, March 2023.

[47] Noah Shinn, Beck Labash, and Ashwin Gopinath. Reflexion: an autonomous agent with dynamic memory and self-reflection. March 2023.

[48] Vered Shwartz, Peter West, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Unsupervised commonsense question answering with self-talk. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4615–4629, Online, November 2020. Association for Computational Linguistics.

[49] Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, Agnieszka Kluska, Aitor Lewkowycz, Akshat Agarwal, Alethea Power, Alex Ray, Alex Warstadt, Alexander W Kocurek, Ali Safaya, Ali Tazarv, Alice Xiang, Alicia Parrish, Allen Nie, Aman Hussain, Amanda Askell, Amanda Dsouza, Ambrose Slone, Ameet Rahane, Anantharaman S Iyer, Anders Andreassen, Andrea Madotto, Andrea Santilli, Andreas Stuhlmüller, Andrew Dai, Andrew La, Andrew Lampinen, Andy Zou, Angela Jiang, Angelica Chen, Anh Vuong, Animesh Gupta, Anna Gottardi, Antonio Norelli, Anu Venkatesh, Arash Gholamidavoodi, Arfa Tabassum, Arul Menezes, Arun Kirubarajan, Asher Mullokandov, Ashish Sabharwal, Austin Herrick, Avia Efrat, Aykut Erdem, Ayla Karakaş, B Ryan Roberts, Bao Sheng Loe, Barret Zoph, Bartłomiej Bojanowski, Batuhan Özyurt, Behnam Hedayatnia, Behnam Neyshabur, Benjamin Inden, Benno Stein, Berk Ekmekci, Bill Yuchen Lin, Blake Howald, Cameron Diao, Cameron Dour, Catherine Stinson, Cedrick Argueta, César Ferri Ramírez, Chandan Singh, Charles Rathkopf, Chenlin Meng, Chitta Baral, Chiyu Wu, Chris Callison-Burch, Chris Waites, Christian Voigt, Christopher D Manning, Christopher Potts, Cindy Ramirez, Clara E Rivera, Clemencia Siro, Colin Raffel, Courtney Ashcraft, Cristina Garbacea, Damien Sileo, Dan Garrette, Dan Hendrycks,

Dan Kilman, Dan Roth, Daniel Freeman, Daniel Khashabi, Daniel Levy, Daniel Moseguí González, Danielle Perszyk, Danny Hernandez, Danqi Chen, Daphne Ippolito, Dar Gilboa, David Dohan, David Drakard, David Jurgens, Debajyoti Datta, Deep Ganguli, Denis Emelin, Denis Kleyko, Deniz Yuret, Derek Chen, Derek Tam, Dieuwke Hupkes, Diganta Misra, Dilyar Buzan, Dimitri Coelho Mollo, Diyi Yang, Dong-Ho Lee, Ekaterina Shutova, Ekin Dogus Cubuk, Elad Segal, Eleanor Hagerman, Elizabeth Barnes, Elizabeth Donoway, Ellie Pavlick, Emanuele Rodola, Emma Lam, Eric Chu, Eric Tang, Erkut Erdem, Ernie Chang, Ethan A Chi, Ethan Dyer, Ethan Jerzak, Ethan Kim, Eunice Engefu Manyasi, Evgenii Zheltonozhskii, Fanyue Xia, Fatemeh Siar, Fernando Martínez-Plumed, Francesca Happé, Francois Chollet, Frieda Rong, Gaurav Mishra, Genta Indra Winata, Gerard de Melo, Germán Kruszewski, Giambattista Parascandolo, Giorgio Mariani, Gloria Wang, Gonzalo Jaimovitch-López, Gregor Betz, Guy Gur-Ari, Hana Galijasevic, Hannah Kim, Hannah Rashkin, Hannaneh Hajishirzi, Harsh Mehta, Hayden Bogar, Henry Shevlin, Hinrich Schütze, Hiromu Yakura, Hongming Zhang, Hugh Mee Wong, Ian Ng, Isaac Noble, Jaap Jumelet, Jack Geissinger, Jackson Kernion, Jacob Hilton, Jaehoon Lee, Jaime Fernández Fisac, James B Simon, James Koppel, James Zheng, James Zou, Jan Kocoń, Jana Thompson, Jared Kaplan, Jarema Radom, Jascha Sohl-Dickstein, Jason Phang, Jason Wei, Jason Yosinski, Jekaterina Novikova, Jelle Bosscher, Jennifer Marsh, Jeremy Kim, Jeroen Taal, Jesse Engel, Jesujoba Alabi, Jiacheng Xu, Jiaming Song, Jillian Tang, Joan Waweru, John Burden, John Miller, John U Balis, Jonathan Berant, Jörg Frohberg, Jos Rozen, Jose Hernandez-Orallo, Joseph Boudeman, Joseph Jones, Joshua B Tenenbaum, Joshua S Rule, Joyce Chua, Kamil Kanclerz, Karen Livescu, Karl Krauth, Karthik Gopalakrishnan, Katerina Ignatyeva, Katja Markert, Kaustubh D Dhole, Kevin Gimpel, Kevin Omondi, Kory Mathewson, Kristen Chiafullo, Ksenia Shkaruta, Kumar Shridhar, Kyle McDonell, Kyle Richardson, Laria Reynolds, Leo Gao, Li Zhang, Liam Dugan, Lianhui Qin, Lidia Contreras-Ochando, Louis-Philippe Morency, Luca Moschella, Lucas Lam, Lucy Noble, Ludwig Schmidt, Luheng He, Luis Oliveros Colón, Luke Metz, Lütfi Kerem Şenel, Maarten Bosma, Maarten Sap, Maartje ter Hoeve, Maheen Farooqi, Manaal Faruqui, Mantas Mazeika, Marco Baturan, Marco Marelli, Marco Maru, Maria Jose Ramírez Quintana, Marie Tolkiehn, Mario Giulianelli, Martha Lewis, Martin Potthast, Matthew L Leavitt, Matthias Hagen, Mátyás Schubert, Medina Orduna Baitemirova, Melody Arnaud, Melvin McElrath, Michael A Yee, Michael Cohen, Michael Gu, Michael Ivanitskiy, Michael Starritt, Michael Strube, Michał Swędrowski, Michele Bevilacqua, Michihiro Yasunaga, Mihir Kale, Mike Cain, Mimee Xu, Mirac Suzgun, Mo Tiwari, Mohit Bansal, Moin Aminnaseri, Mor Geva, Mozhdeh Gheini, Varma T Mukund, Nanyun Peng, Nathan Chi, Nayeon Lee, Neta Gur-Ari Krakover, Nicholas Cameron, Nicholas Roberts, Nick Doiron, Nikita Nangia, Niklas Deckers, Niklas Muennighoff, Nitish Shirish Keskar, Niveditha S Iyer, Noah Constant, Noah Fiedel, Nuan Wen, Oliver Zhang, Omar Agha, Omar Elbaghdadi, Omer Levy, Owain Evans, Pablo Antonio Moreno Casares, Parth Doshi, Pascale Fung, Paul Pu Liang, Paul Vicol, Pegah Alipoormolabashi, Peiyuan Liao, Percy Liang, Peter Chang, Peter Eckersley, Phu Mon Htut, Pinyu Hwang, Piotr Miłkowski, Piyush Patil, Pouya Pezeshkpour, Priti Oli, Qiaozhu Mei, Qing Lyu, Qinlang Chen, Rabin Banjade, Rachel Etta Rudolph, Raefer Gabriel, Rahel Habacker, Ramón Risco Delgado, Raphaël Millière, Rhythm Garg, Richard Barnes, Rif A Saurous, Riku Arakawa, Robbe Raymaekers, Robert Frank, Rohan Sikand, Roman Novak, Roman Sitelew, Ronan LeBras, Rosanne Liu, Rowan Jacobs, Rui Zhang, Ruslan Salakhutdinov, Ryan Chi, Ryan Lee, Ryan Stovall, Ryan Teehan, Rylan Yang, Sahib Singh, Saif M Mohammad, Sajant Anand, Sam Dillavou, Sam Shleifer, Sam Wiseman, Samuel Gruetter, Samuel R Bowman, Samuel S Schoenholz, Sanghyun Han, Sanjeev Kwatra, Sarah A Rous, Sarik Ghazarian, Sayan Ghosh, Sean Casey, Sebastian Bischoff, Sebastian Gehrmann, Sebastian Schuster, Sepideh Sadeghi, Shadi Hamdan, Sharon Zhou, Shashank Srivastava, Sherry Shi, Shikhar Singh, Shima Asaadi, Shixiang Shane Gu, Shubh Pachchigar, Shubham Toshniwal, Shyam Upadhyay, Shyamolima, Debnath, Siamak Shakeri, Simon Thormeyer, Simone Melzi, Siva Reddy, Sneha Priscilla Makini, Soo-Hwan Lee, Spencer Torene, Sriharsha Hatwar, Stanislas Dehaene, Stefan Divic, Stefano Ermon, Stella Biderman, Stephanie Lin, Stephen Prasad, Steven T Piantadosi, Stuart M Shieber, Summer Misherghi, Svetlana Kiritchenko, Swaroop Mishra, Tal Linzen, Tal Schuster, Tao Li, Tao Yu, Tariq Ali, Tatsu Hashimoto, Te-Lin Wu, Théo Desbordes, Theodore Rothschild, Thomas Phan, Tianle Wang, Tiberius Nkinyili, Timo Schick, Timofei Kornev, Timothy Telleen-Lawton, Titus Tunduny, Tobias Gerstenberg, Trenton Chang, Trishala Neeraj, Tushar Khot, Tyler Shultz, Uri Shaham, Vedant Misra, Vera Demberg, Victoria Nyamai, Vikas Raunak, Vinay Ramasesh, Vinay Uday Prabhu, Vishakh Padmakumar, Vivek Srikumar, William Fedus, William Saunders, William Zhang, Wout Vossen, Xiang Ren, Xiaoyu

Tong, Xinran Zhao, Xinyi Wu, Xudong Shen, Yadollah Yaghoobzadeh, Yair Lakretz, Yangqiu Song, Yasaman Bahri, Yejin Choi, Yichi Yang, Yiding Hao, Yifu Chen, Yonatan Belinkov, Yu Hou, Yufang Hou, Yuntao Bai, Zachary Seid, Zhuoye Zhao, Zijian Wang, Zijie J Wang, Zirui Wang, and Ziyi Wu. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. June 2022.

[50] K L Stachenfeld, M Botvinick, and others. Design principles of the hippocampal cognitive map. *Adv. Neural Inf. Process. Syst.*, 2014.

[51] Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT Press, November 2018.

[52] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following LLaMA model, 2023.

[53] Rita Morais Tavares, Avi Mendelsohn, Yael Grossman, Christian Hamilton Williams, Matthew Shapiro, Yaacov Trope, and Daniela Schiller. A map for social navigation in the human brain. *Neuron*, 87(1):231–243, July 2015.

[54] Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. LaMDA: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*, 2022.

[55] E C Tolman. Cognitive maps in rats and men. *Psychol. Rev.*, 55(4):189–208, July 1948.

[56] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[57] Tomer Ullman. Large language models fail on trivial alterations to Theory-of-Mind tasks. February 2023.

[58] Karthik Valmeekam, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. Large language models still can't plan (a benchmark for LLMs on planning and reasoning about change). June 2022.

[59] Marlieke T R van Kesteren, Sarah F Beul, Atsuko Takashima, Richard N Henson, Dirk J Ruiter, and Guillén Fernández. Differential roles for medial prefrontal and medial temporal cortices in schema-dependent encoding: from congruent to incongruent. *Neuropsychologia*, 51(12):2352–2359, October 2013.

[60] Marlieke Tina Renée van Kesteren and Martijn Meeter. How to optimize knowledge construction in the brain. *NPJ Sci Learn*, 5:5, May 2020.

[61] Yi Wan, Ali Rahimi-Kalahroudi, Janarthanan Rajendran, Ida Momennejad, Sarath Chandar, and Harm van Seijen. Towards evaluating adaptivity of Model-Based reinforcement learning methods. April 2022.

[62] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, and Denny Zhou. Rationale-augmented ensembles in language models. *arXiv preprint arXiv:2207.00747*, 2022.

[63] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

[64] Taylor Webb, Keith J Holyoak, and Hongjing Lu. Emergent analogical reasoning in large language models. December 2022.

[65] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022.

[66] James C R Whittington, Joseph Warren, and Timothy E J Behrens. Relating transformers to models and neural representations of the hippocampal formation. December 2021.

[67] Eunice Yiu, Eliza Kosoy, and Alison Gopnik. Imitation versus innovation: What children can do that large language and language-and-vision models cannot (yet)? May 2023.

[68] Eric Zelikman, Yuhuai Wu, and Noah D Goodman. STaR: Bootstrapping reasoning with reasoning. *arXiv preprint arXiv:2203.14465*, 2022.

[69] Denny Zhou, Nathanael Scharli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Olivier Bousquet, Quoc Le, and Ed Chi. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*, 2022.

Evaluating Cognitive Maps in Large Language Models: No Emergent Planning (Supplementary Materials)

Anonymous Author(s)
Affiliation
Address
email

1 Supplementary Experiment: Systematic graph explorations

To systematically evaluate GPT-4's planning or graph traversal failure modes, we created a three-block community graph structures where each block contains five vertices. Using this approach, we vary the connection density within each community block and ask GPT-4 to perform reasoning tasks over each permutation of the graph structure as block density is varied. For the graph community block model, example graphs are shown in Figure 1 with the community graphs starting as simple line graphs on the left - representing the sparsest level of connectivity. We then create a new edge within each block for each iteration of the experiment until each community block forms a clique structure as seen on the right of Figure 1. To measure performance, the LLM is asked to assign partitions for each vertex such as to maximize each graph's modularity. Modularity is chosen as the task as it requires a *non-trivial understanding of the graph* beyond the local network of any single vertex in order to detect the boundaries between communities. The LLM's vertex assignment is then compared to the vertex assignments obtained from a Leiden [**?**] modularity maximization process. The results are compared using Adjusted Rand Index (ARI), which gives a similarity score between the actual modularity-maximized partitioning scheme and the observed partitioning that the LLM returned. ARI scores closer to zero represent poor performance by the LLM and ARI scores reaching 1.0 represent performance matching Leiden. This is performed for temperatures 0.05, 0.5, 0.95 and TopP 0.05, 0.5, 0.95. Each configured test is executed thirty times through the GPT-4 chat completion API.

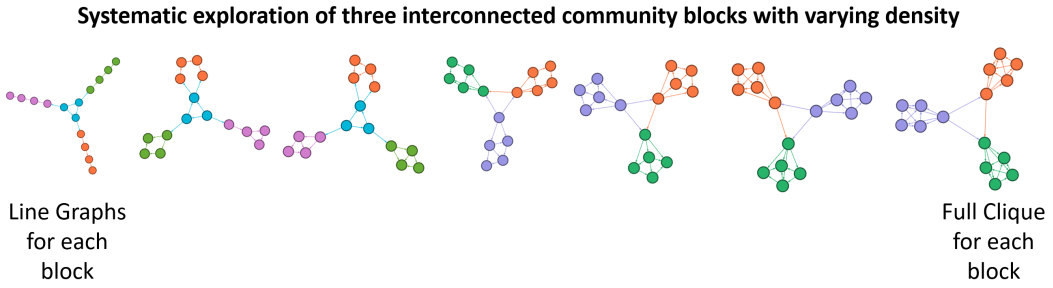

Figure 1: **Systematic investigation of line graphs to full-clique structures.** Each graph is composed of three blocks, where each block is its own sub-graph structure with five interconnected vertices on which we vary the edge density. Each block has a bridge node linked to other blocks' bridge nodes. The left graph displays the sparsest density with a line graph for each block, while the right graph shows the densest blocks with each forming a fully interconnected clique. Vertex partition assignments are color-coded via Leiden modularity maximization.

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

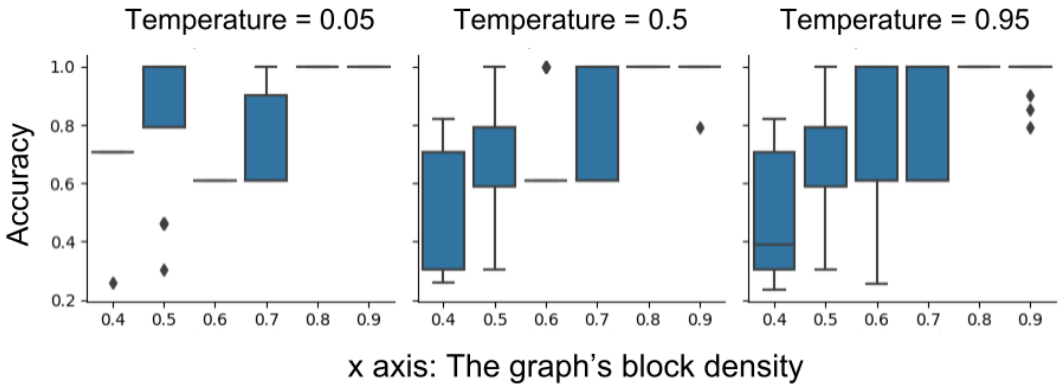

Figure 2: **Supplementary experiment results.** Systematic investigation of line graphs to full-clique structures. y-axis: accuracy of the LLM as measured using the Adjusted Rand Index (ARI) between a maximal modularity partitioning from the LLM as compared to observed maximal modularity via Leiden. An ARI of 1.0 indicates the LLM matched Leiden. x-axis: the density of each block, 0.4 represents a line graph of the five vertices and 1.0 represents a fully connected clique structure. Temperature: 0.05, 0.5, 0.95,TopP: 0.95.

1.1 Supplementary Experiment Evaluation: Evaluating the systematic effect of graph structure

Figure 2 shows the result of systematic evaluation over the community based graph structures using GPT-4. The y-axis measures how well the LLM performed (higher is better) and the x-axis measures the density of each of the community block structure. Outlier points are added directly to each plot and each panel shows how performance varies with temperature. These findings suggest that the LLM performs more poorly in the sparse community structures, but better as the edge density increases. Additionally, a higher temperature results in poorer performance with higher variance. These results may appear in contrast to the results shown in Experiment 1, where LLMs show apparent success in *local traversal* of a small graph. However, as the task changed from local traversal (Experiment 1) towards optimization and reasoning of a non-local graph structure (Supplementary Experiment), we observe the LLMs fail when the structures are sparse. This points to the LLM's struggle to reason over local neighborhood and community structures, and is consistent with failure on community graphs as demonstrated in Experiment 1. Moreover, these findings show how *the LLMs' behavior is not consistent* over community structures vs. simple traversals, varying across overall sparsity. This is in line with the hypothesis that LLMs lack functional understanding of underlying structures of the problems, which may contribute to their failure in multi-step planning.

In order to better understand failure modes, this experiment systematically evaluates the observed poor performance of GPT-4 on community graphs on gradually more dense community graphs.

1.2 Supplementary Experiment: Evaluating the systematic effect of graph structure as TopP is varied

Below are the graphs for each TopP configuration:Systematic graph explorations and the variance of TopP

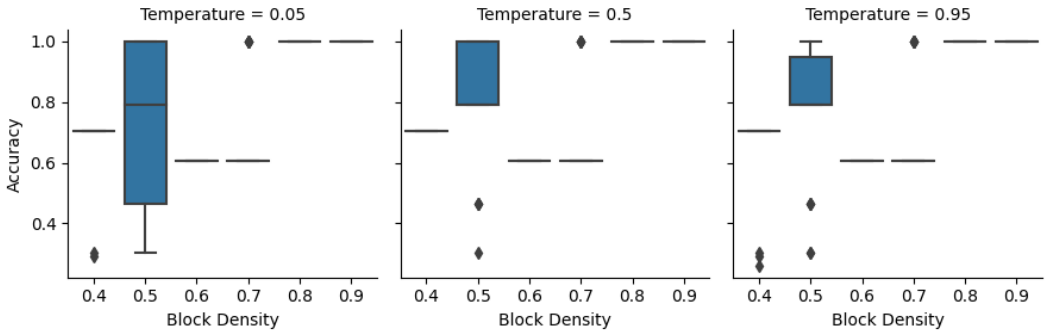

Figure 3: Results of systematic investigation of line graphs to full-clique structures. TopP = 0.05.

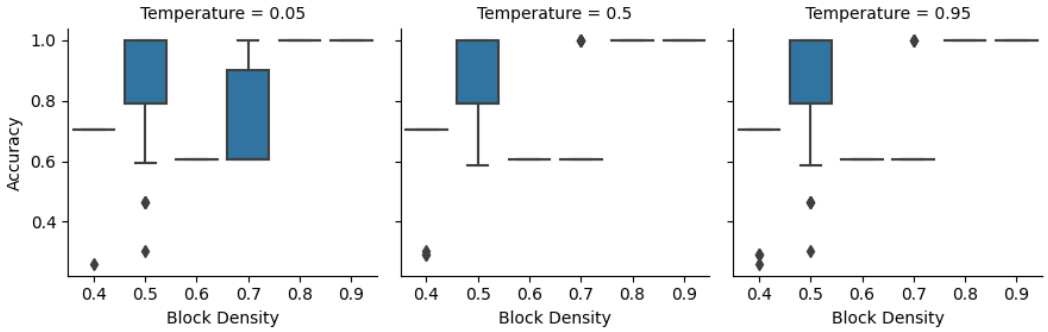

Figure 4: Results of systematic investigation of line graphs to full-clique structures. TopP = 0.5.

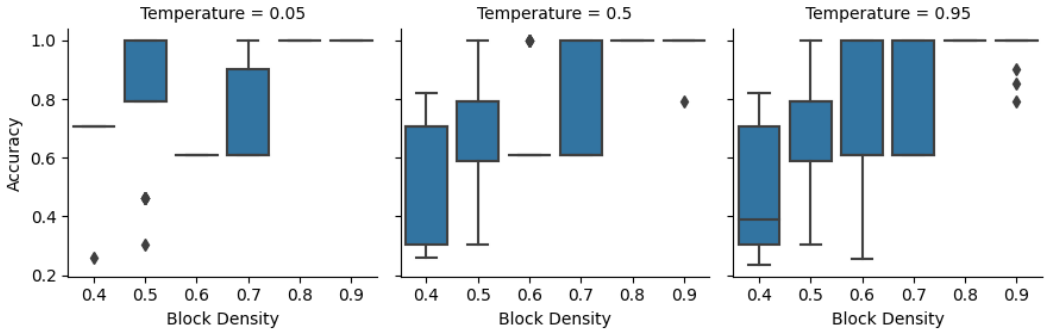

Figure 5: Results of systematic investigation of line graphs to full-clique structures. TopP = 0.95.

1.3 Supplementary Experiment: Prompt templates

Figure 6 shows the prompt templates that were used in this supplementary experiment.

```
prompt_id,calculation,prompt_template,prompt,token_count,response
prompt_01,modularity,"Given the following comma delimited edge list of source and target nodes in a graph, assign each node a cluster as to maximize the
modularity of the graph. Provide the results in JSON format with the key representing the node and the value representing the community.

Example:
Source,Target
1,2
2,1
3,4
4,3

Answer:
{
    ""1"": 1,
    ""2"": 1,
    ""3"": 2,
    ""4"": 2
}

Graph to Analyze
========
Source,Target
{EDGE_LIST}
========",,,
```

Figure 6: Prompt template for maximizing modularity

2 CoT prompts used for BFS and DFS

We have used quite general and simple description of graph traversal algorithms in the prompt to measure how much the LLM can leverage the information in the prompt. The prompts that we have used for BFS and DFS are as follows and they are directly appended to the existing prompt for each task.

- **BFS**: "*Think carefully before you respond. You can try using Breadth-first search (BFS), it is a graph traversal algorithm that visits all the vertices of a graph in breadth-first order, starting from a given source vertex. In BFS, vertices are visited in layers, where the vertices at distance 1 from the source vertex are visited first, followed by the vertices at distance 2, and so on. BFS uses a queue data structure to keep track of the vertices to be visited, and it ensures that no vertex is visited more than once. BFS is useful for finding the shortest path between two vertices in an unweighted graph, or for exploring all the vertices in a graph.\endofprompt*|"

- **DFS**: "*Think carefully before you respond. You can try using Depth-first search (DFS), it is a graph traversal algorithm that visits all the vertices of a graph in depth-first order, starting from a given source vertex. In DFS, the algorithm traverses as far as possible along each branch before backtracking. DFS uses a stack data structure to keep track of the vertices to be visited, and it ensures that all vertices connected to a visited vertex are explored before backtracking. DFS is useful for finding cycles in a graph, for exploring all the vertices in a graph, or for finding a path between two vertices. However, unlike BFS, DFS does not guarantee that the shortest path is found.\endofprompt*|"

3 Brief descriptions of the task conditions applied to varying graphs and domains

Table 1: Brief descriptions of the task conditions applied to varying graphs and domains

Condition	Description	Group
valuePath	The optimal solution is to find the optimal policy, or shortest path, which yields the highest reward	**Traversal**
1stepPath	The optimal solution is a 1-hop policy, i.e., goal is adjacent to the starting state	
2stepPath	The optimal solution is a 2-step policy	
3stepPath	The optimal solution is a 3-step policy	
nstepPath	The optimal solution is an n-step policy, where max n is the diameter of the graph (longest shortest path)	
rewardReval	Upon a local change in the *reward structure*, the goal has changed and the optimal solution requires finding a new path	**RewReval**
policyReval	Upon a local change in the *reward structure*, the optimal solution requires finding a new policy	
transReval	Upon a local change in the *transition structure*, the goal is the same but the optimal solution requires finding a new policy	**TransReval**
transRevalStochastic	Upon a local change in the *transition structure*, the goal is the same but the optimal solution requires finding a new policy in a stochastic environment	
nonteleShortcut	Upon a change in the graph structure, the optimal solution requires finding a shortcut	**Shortcut**
nonteleShortcutCoT	Upon a change in the graph structure, the optimal solution requires finding a shortcut, an additional CoT prompt is given	
teleShortcut	Upon a local change in the *transition structure*, the optimal solution requires finding a shortcut using a teleportation portal	
teleShortcutCoT	Upon a local change in the *graph or transition structure*, the optimal solution requires finding a shortcut using a teleportation portal, an additional CoT prompt is given	
nonteleDetour	Upon a change in the graph structure, the optimal solution requires finding a detour	**Detour**
teleDetour	Upon a local change in the *transition structure*, the optimal solution requires finding a detour using a teleportation step	

4 Summary of high-level statistical analysis

In the presented study, the "score" of a dialog is the number of correct answers provided by the LLM out of a total number of correct answers possible for that dialog. We modeled the score using a logistic regression approach; the score follows a binomial distribution with a probability parameter determined by the three categorical variables (graph structure, condition, and domain) as well as model and temperature. We our regression model included second and third-order interaction terms between levels of these three terms.

Our initial strategy was to assume that for a particular combination of the three factors (graph structure, condition, and domain), the conjunction of model and temperature could be likened to a 'subject' in a repeated measures analysis. With the inability to set a seed in these LLMs, we posited that each repeated measurement for the engine and temperature variables could be akin to a repeated replicate measure in a longitudinal or panel study, where there would be "within-subject variation" across replicates. We introduced a nested random effect (temperature nested within model) to the linear component of a linear regression model. Note that we did not have an equal number of replicates across each combination of graph structure, condition, domain, model, and temperature (the minimum number of replicates for a combination was 1, the maximum was 30, and the mean was 7.1). However, the logistic regression approach is robust to replicate imbalance.

We used a two-step fitting process. In the first step, we used elastic net to fit the model using the R package glmnet. We relied on elastic net's mix of L1 and L2 regularization to address cases of LLMs where we collected less data, and to address the multicollinearity introduced by the interaction terms. The parameter estimates are shown in Table 2. Next, we refit a new model using non-regularized logistic regression on the predictors with non-zero coefficient estimates in the first model, and used this model to generate the analysis of variance (deviance) table in Table **??**. Deviance is a measure of goodness of fit; it quantifies the discrepancy between the observed scores and the scores predicted by the model. Each row in the Chi-squared statistic column quantifies the reduction in deviance by adding the categorical variables associated with the term in that row, given the variables from the previous rows are included in the model.

5 Experiment 1, statics of results: Repeated measures comparison of planning across LLMs

We evaluated out-of-the-box emergent or native ability of different LLMs on the cognitive map tasks. Table 2 shows the statistical analysis highlighting the contributions of each factor to regression model's

Table 2: Parameter estimates from the regularized logistic regression model. Baselines are condition:traversal, graph:n7line, domain:ordRooms, LLM:replicate-alpaca-7b, temp:0. NA values indicate the data was not sufficient to fit the parameters.

	factor	level	estimate	odds multiple	p value
1	(Intercept)	(Intercept)	0.85	2.33	<0.001
2	LLM	bard	-0.41	0.66	0.01
3	LLM	cohere-xlarge	-2.25	0.11	<0.001
4	LLM	gpt-35-turbo	-0.40	0.67	<0.001
5	LLM	gpt-4-32k	1.25	3.50	<0.001
6	LLM	replicate-alpaca-7b	-3.57	0.03	<0.001
7	LLM	replicate-llama-13b	-2.67	0.07	<0.001
8	LLM	text-davinci-003	-0.73	0.48	<0.001
9	graph	n13line	-1.14	0.32	<0.001
10	graph	n7tree	-2.59	0.07	<0.001
11	graph	n15star	-1.47	0.23	<0.001
12	graph	n21star	-1.78	0.17	<0.001
13	graph	n16cluster	-0.62	0.54	<0.001
14	domain	socialTies	0.58	1.78	<0.001
15	domain	unordSpatial	-1.25	0.29	<0.001
16	temperature	0.5	-0.00	1.00	0.96
17	temperature	1	-0.06	0.95	0.12
18	condition	Detour	-2.35	0.10	<0.001
19	condition	RewReval	-2.56	0.08	<0.001
20	condition	Shortcut	-2.01	0.13	<0.001
21	condition	TransReval	-2.23	0.11	<0.001
22	model and temp	gpt-4-32k.0.5	0.04	1.04	0.52
23	model and temp	gpt-4-32k.1	0.05	1.05	0.37
24	graph and domain	n7line & ordRooms	2.11	8.27	<0.001
25	graph and domain	n7tree & ordRooms	1.01	2.75	<0.001
26	graph and domain	n7line & socialTies	0.77	2.16	<0.001
27	graph and domain	n7line & unordSpatial	NA	NA	NA
28	graph and condition	n7line & Detour	1.95	7.02	<0.001
29	graph and condition	n13line & RewReval	3.18	23.99	<0.001
30	graph and condition	n16cluster & RewReval	1.46	4.30	<0.001
31	graph and condition	n7line & Shortcut	1.66	5.28	<0.001
32	graph and condition	n13line & Traversal	2.06	7.88	<0.001
33	graph and condition	n16cluster & Traversal	0.13	1.14	0.19
34	graph and condition	n7line & Traversal	1.32	3.76	<0.001
35	graph and condition	n7tree & Traversal	1.97	7.17	<0.001
36	domain and condition	socialTies & Shortcut	-0.02	0.98	0.90
37	domain and condition	ordRooms & Traversal	-0.48	0.62	<0.001
38	domain and condition	socialTies & Traversal	-0.99	0.37	<0.001
39	graph, domain, condition	n16cluster & ordRooms & Detour	0.70	2.02	<0.001
40	graph, domain, condition	n7line & ordRooms & Detour	0.50	1.65	0.10
41	graph, domain, condition	n7tree & ordRooms & Detour	2.39	10.94	<0.001
42	graph, domain, condition	n15star & socialTies & Detour	1.30	3.69	<0.001
43	graph, domain, condition	n21star & socialTies & Detour	1.01	2.73	<0.001
44	graph, domain, condition	n7line & socialTies & Detour	-1.06	0.35	<0.001
45	graph, domain, condition	n15star & unordSpatial & Detour	1.78	5.92	<0.001
46	graph, domain, condition	n21star & unordSpatial & Detour	2.24	9.40	<0.001
47	graph, domain, condition	n13line & ordRooms & RewReval	-0.57	0.57	0.01
48	graph, domain, condition	n7tree & ordRooms & RewReval	2.58	13.20	<0.001
49	graph, domain, condition	n13line & socialTies & RewReval	-0.20	0.82	0.38
50	graph, domain, condition	n15star & socialTies & RewReval	2.58	13.25	<0.001
51	graph, domain, condition	n16cluster & socialTies & RewReval	0.80	2.23	<0.001
52	graph, domain, condition	n15star & unordSpatial & RewReval	3.61	36.98	<0.001
53	graph, domain, condition	n16cluster & unordSpatial & RewReval	2.07	7.94	<0.001
54	graph, domain, condition	n7line & unordSpatial & RewReval	2.58	13.21	<0.001
55	graph, domain, condition	n7line & ordRooms & Shortcut	-1.76	0.17	<0.001
56	graph, domain, condition	n21star & socialTies & Shortcut	0.91	2.48	<0.001
57	graph, domain, condition	n7line & socialTies & Shortcut	-0.48	0.62	0.06
58	graph, domain, condition	n7tree & socialTies & Shortcut	3.59	36.40	<0.001
59	graph, domain, condition	n15star & unordSpatial & Shortcut	1.95	7.05	<0.001
60	graph, domain, condition	n16cluster & unordSpatial & Shortcut	1.28	3.59	<0.001
61	graph, domain, condition	n13line & ordRooms & TransReval	2.31	10.12	<0.001
62	graph, domain, condition	n15star & ordRooms & TransReval	1.04	2.82	<0.001
63	graph, domain, condition	n7tree & ordRooms & TransReval	2.84	17.13	<0.001
64	graph, domain, condition	n7tree & socialTies & TransReval	3.20	24.49	<0.001
65	graph, domain, condition	n13line & unordSpatial & TransReval	3.37	29.19	<0.001
66	graph, domain, condition	n16cluster & unordSpatial & TransReval	1.78	5.92	<0.001
67	graph, domain, condition	n7line & unordSpatial & TransReval	2.65	14.15	<0.001
68	graph, domain, condition	n13line & ordRooms & Traversal	1.70	5.47	<0.001
69	graph, domain, condition	n7line & ordRooms & Traversal	-0.67	0.51	0.01
70	graph, domain, condition	n13line & socialTies & Traversal	0.77	2.15	<0.001
71	graph, domain, condition	n16cluster & socialTies & Traversal	-0.02	0.98	0.87
72	graph, domain, condition	n7line & socialTies & Traversal	NA	NA	NA
73	graph, domain, condition	n21star & unordSpatial & Traversal	0.65	1.91	<0.001
74	graph, domain, condition	n7tree & unordSpatial & Traversal	0.72	2.06	<0.001

fit of LLM model performance. The magnitude of chi-square test statistics indicate contribution to overall model fit. Figure 3 compares the performance of all LLMs across all latent graph structures. Table 3 shows mean and standard error for planning performance across tasks and LLMs.

The results in Table 2 indicate that the model engine ($\chi^2(11) = 689.36$, p < .001.), graph ($\chi^2(11) = 7247.30$, p < .001.), condition ($\chi^2(11) = 1475.03$, p < .001.), and domain ($\chi^2(11) = 304.06$, p < .001.) each yielded significant chi-squared statistics. This means that not only did different LLMs performed differently, but each LLM's performance varied as a result of varying graphs, domains, and conditions. Conversely, the temperature showed a non-significant chi-squared statistic ($\chi^2(11) = 1.01, p = .6022$) and the interaction between the model and temperature was also

non-significant ($\chi^2(11) = 4.65, p = .997$). Noteworthy, the interactions among graph-domain, graph-condition, domain-condition, and graph-domain-condition were all significant (all p's < .001). The interactions among graph-domain ($\chi^2(11) = 689.36, p < .001$), graph-condition ($\chi^2(50) = 1392.82, p < .001$), domain-condition ($\chi^2(39) = 524.48, p < .001$), and graph-domain-condition ($\chi^2(108) = 1002.93, p < .001$) were all significant.

In summary, while the 'temperature' and the interaction of 'model' and 'temperature' do not show significant effects in Experiment 1 (but show difference in Experiments 2 and 3), all other factors and their interactions significantly contribute to the variations in the dependent variable. Considering both individual and interactive effects, this finding shows that LLM performance on cognitive map and planning tasks was not robust to the graph structure of the problems, the domain, or the task conditions, and it also varied across models (see Tables 2 and 3 and Figure 3 .