Language Models as Narrative Planning Heuristics

BY LASANTHA SENANAYAKE AND STEPHEN G. WARE

PRESENTED BY EVAN DAMRON

Introduction

- Large Language Models (LLMs) have been trained on vast quantities of textual data, including narratives.

- However, they aren't designed to maintain coherence on long sequences of actions, so their performance is unreliable.

- This paper evaluates whether LLMs can provide accurate estimates of story size and content, paving the way to eventually use LLMs as a heuristic guide in state-space searches.

Background

- Sabre is a state space narrative planner incorporating intentionality and belief. A solution achieves the author's goal and only includes explained actions.
- To navigate the large search space of narrative planning problems, it is crucial for planners like Sabre to have a fast and accurate heuristic.
- The intuition behind this paper is that LLMs will be able to reason about character intentions and beliefs, so it will suggest more easily explainable actions than classical heuristics:
  - h^{+}: HSP heuristic, where cost of a conjunction is the sum of conjuncts.
  - h^{max}: HSP heuristic, where cost of a conjunction is the max of conjuncts.
  - h^{rp}: Fast Forward heuristic, builds a plan graph and solves the relaxed problem. This heuristic returns a plan (or a narrative), just like the LLM.

Data Collection Phase

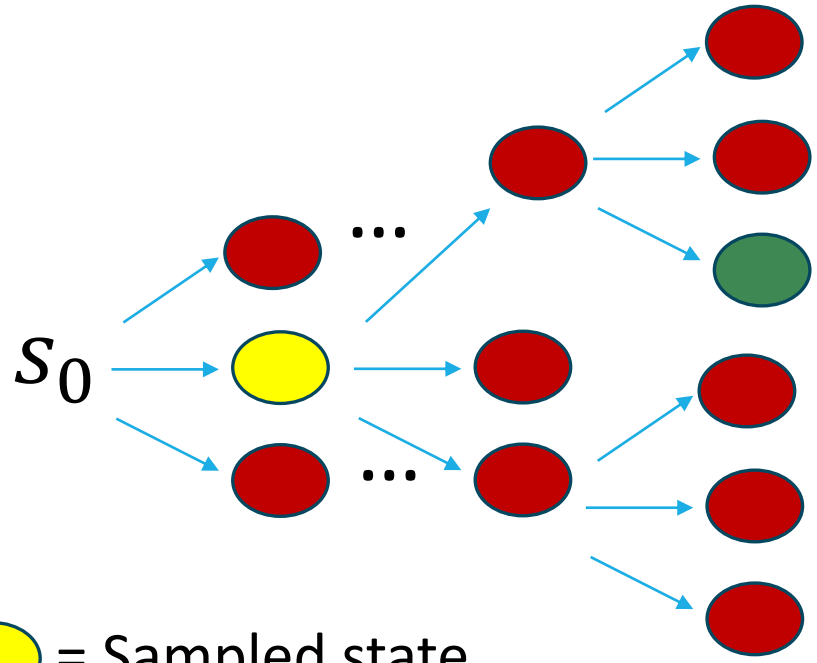
- To evaluate a heuristic, you need to know the actual distance from a given state to the goal state.

- To do this, the authors ran breadth first search on a collection of narrative planning problems, yielding the shortest solutions.

- Each problem was solved to the maximum depth that could be reached in three days, running on a computer with an Intel Xeon 4.1 GHz processor and 512 GB RAM.

Example Diagram

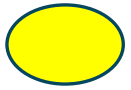
Setup phase BFS



= Sampled state

= Solution state

Heuristic calculations

- Send  to OpenAI's GPT-4o mini, prompting it to finish the narrative.
- Compare the number of actions returned by the LLM to the actual distance (2).
- Compare the actions included by the LLM to the ones in the actual plan.

Prompt Design

I will describe a setting and the first part of a story. Your job is to complete the story to ensure it has a specific ending. → Task

There are two locations in this story: the port and the island. There is one item in this story: some treasure. There are two characters in this story. Hawkins is a boy who wants the treasure. Silver is a pirate who wants the treasure. There are four kinds of actions characters can take in the story. If the treasure is buried on the island, Hawkins can spread a rumor that will make Silver believe the treasure is buried on the island. Hawkins and Silver can work together to sail a ship from the port to the island. If the treasure is buried on the island, Hawkins can dig up the treasure. A character can take the treasure once it has been dug up. → Problem

These events have already happened in the story: Hawkins spreads a rumor that the treasure is buried on the island. Hawkins and Silver sail from Port Royal to the island. Hawkins digs up the treasure. → Previous actions

This is the current situation after those events: Hawkins is at the island. Silver is at the island. The treasure is at the island. Silver believes that the treasure is at the island. → Current state

Complete the story using only these locations, items, characters, and actions. Do not invent new locations, items, characters, or actions. Characters should only take actions that help to achieve their goals, and the story should only include actions which are necessary to achieve the ending. → Instructions

Give me the shortest story where Hawkins achieves their goal. → Goal

Explain why each action is in the story. After the explanation of the whole story, give a JSON object with the final plan. The JSON should include an array called 'plan' with the sequence of actions (as a string) taken to achieve the goal. Example format: {plan: ['action1', 'action2', 'action3']}. → Formatting Instructions

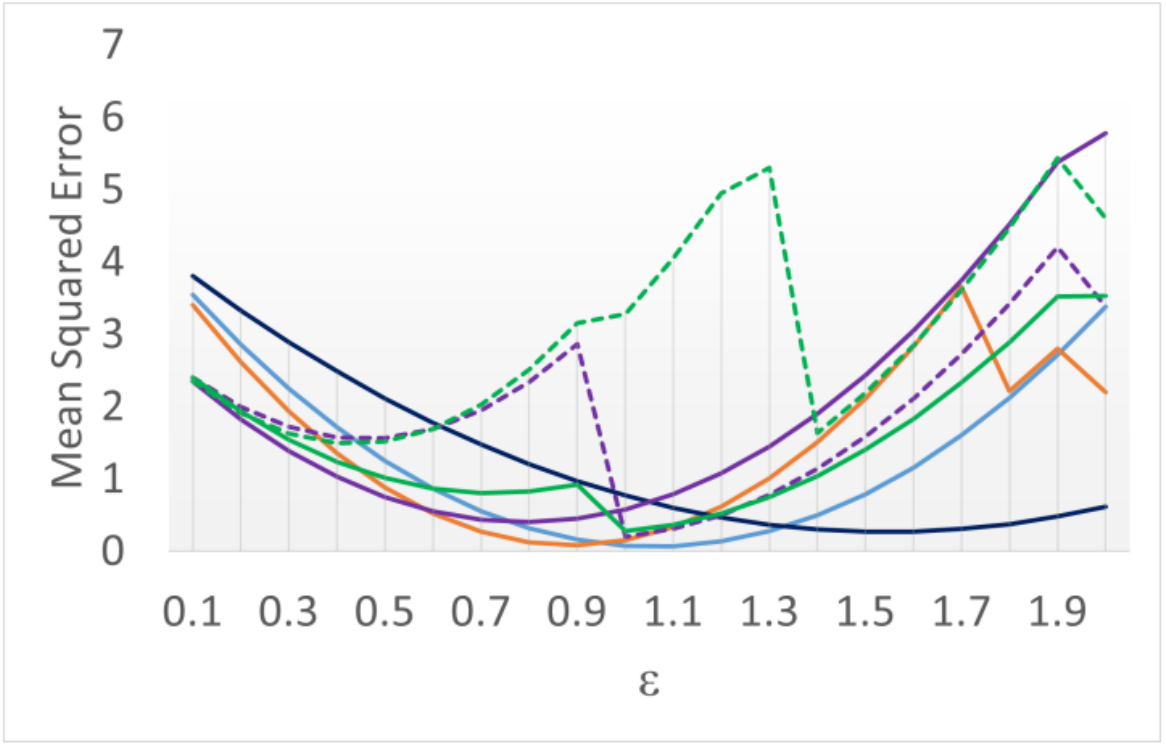
Prompt Variations

- Prompt with Natural Language: Translate from Sabre syntax.
  - location(Gargax) = Cave becomes Gargax is at the Cave.
  - believes(Talia, alive(Gargax)) = True becomes Talia believes Gargax is alive.
- Prompt with Syntax: Create prompt directly with the Sabre syntax.
  - Requires less special-purpose coding.
- Both approaches were also used with a suggested maximum length of the plan, by appending this to the prompt:
  - “While keeping the plan complete, a smaller plan is preferred. Suggested maximum length of the plan: {SUGGESTED_PLAN_LENGTH}.”
- SUGGESTED_PLAN_LENGTH = (MAX_PLAN_LENGTH) - (NUM_ACTIONS_DONE)

Evaluation

- They use the OpenAI LLM GPT-4o mini, sampling 1,000 states for each problem (only sampling states that were not a goal state).
- To parse the results, they relied on the text-embedding-ada-002 model. As a preprocessing step, they embed every ground action in the problem. They created an embedding of each action returned from the model, and assume it represents the action it has a minimum cosine distance to.
- They weighted each heuristic by a constant ϵ, which ranges from 0.1 to 2.0 to find the ideal version of it.
- The authors evaluated the accuracy of each heuristic and the quality of the relaxed plans returned by the LLM and h^{rp}.

Heuristic Accuracy



(k) Treasure

LLM Natural Heuristic with Limits

LLM Syntax Heuristic with Limits

LLM Natural Heuristic

LLM Syntax Heuristic

Relaxed Plan Heuristc

h_{+} Heuristc

h_{max} Heuristc

Heuristic Accuracy 2

Table 2: Minimum Mean Squared Error Values Fine-Tuning ϵ for Each Problem

Problem	h^{max}	h^{+}	h^{rp}	*hSyntax*	*hSyntaxLimit*	*hNatural*	*hNaturalLimit*
Bribery	**0.0000**	**0.0000**	**0.0000**	0.8789	0.4537	0.7486	0.7066
Deer Hunter	0.0511	**0.2685**	0.5022	0.6235	0.6020	0.6021	0.6868
Fantasy	**0.0070**	0.4770	0.2648	0.4922	0.5147	0.3768	0.5605
Gramma	0.1396	**0.0750**	0.4787	0.6527	0.5070	0.6229	0.5360
Hospital	0.0926	**0.0380**	0.5365	0.6539	0.5638	0.6298	0.5941
Jailbreak	0.3118	**0.1520**	0.2935	1.7975	1.7661	1.8149	1.9105
Lovers	**0.1127**	0.1333	0.3651	0.6428	0.7306	0.4924	0.6858
Raiders	0.7329	**0.1648**	0.4981	1.9185	1.5562	1.6951	0.9366
Secret Agent	1.3323	1.4759	2.5128	5.2686	**0.7517**	3.6071	0.7959
Space	0.0968	**0.0661**	0.4546	0.3425	0.4160	0.3706	0.4302
Treasure	**0.0685**	0.0846	0.2692	0.2000	0.4042	1.4905	0.2778

Heuristic Predictions of Plan Content 2

Table 3: Percent Accuracy of Heuristics in Predicting the First Correct Action

Problem	h^{rp}	*hSyn*	*hSynLim*	*hNat*	*hNatLim*
Bribery	**45.7**	17.1	37.1	14.3	14.3
Deer Hunter	**87.1**	69.3	64.3	35.7	30.3
Fantasy	**66.9**	45.4	41.7	26.8	30.3
Gramma	19.2	**24.9**	22.9	16	8.7
Hospital	**49.6**	1.5	2.1	8.7	9.2
Jailbreak	**61.5**	35.8	41.6	29.5	23.8
Lovers	**36.6**	9.1	12.7	11	13.6
Raiders	**41.5**	24.5	28.3	13.2	18.9
Secret Agent	**61.2**	34.7	49.0	26.5	30.6
Space	34.4	46.1	**52.9**	41.4	45.2
Treasure	36.8	**42.1**	36.8	36.8	**42.1**

Table 4: Frequency of Correct Actions Appearing Anywhere in Heuristic-Generated Plans

Problem	h^{rp}	*hSyn*	*hSynLim*	*hNat*	*hNatLim*
Bribery	**45.7**	25.7	37.1	14.3	14.3
Deer Hunter	**92.6**	78.3	67.9	55.1	39.1
Fantasy	**68.1**	57.5	44.8	39.3	37.1
Gramma	**30.0**	29.1	24.1	24.3	11.9
Hospital	**66.1**	3.3	5.0	16.1	14.0
Jailbreak	**69.9**	64.5	53.1	48.6	33.6
Lovers	**39.4**	14.4	13	20.3	17.2
Raiders	**49.1**	35.9	37.7	24.5	18.9
Secret Agent	**69.4**	42.9	49.0	28.6	30.6
Space	59.2	**82.6**	69.9	80	60.2
Treasure	36.8	42.1	**47.4**	42.1	42.1

Limitations and Conclusions

- The time to query the OpenAI API is a significant bottleneck, preventing using LLM-based heuristics for searching.
- Prompt engineering heavily affects results, but it is not an exact science. Crafting prompts requires significant expertise and effort and is hard to reproduce.
- There is no universal LLM prompt or epsilon value that is best, showing the need to adapt heuristics to each problem.
- The technology is not there yet to consider LLMs the best option for narrative planning heuristics, but they show promise. This paper sets a benchmark for LLM performance, which can be used to evaluate future models.