Hierarchical Logic Graphs for Grounding Goals and Answering Questions

Center for Integrated Cognition

James Kirk

james.kirk@cic.iqmri.org

45th Soar Workshop

May 5, 2025

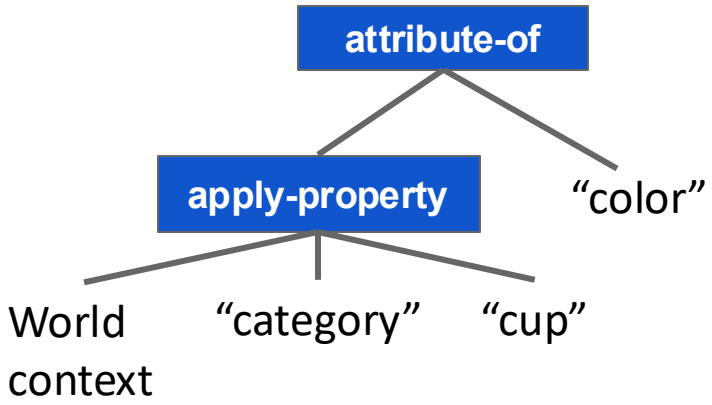
Motivation

- Need to be able to ground and reason over complex goals and questions
- Direct response from LLM are good, but can struggle with handling complex goals and questions which can include
  - involved calculations and spatial reasoning
  - complex combinations of tests and calculations
  - references to unseen/ungroundable objects
  - reasoning over historic information
- Explorations with intermediate soar representations for question answering required creating custom query formats and interpretation code for each type of question
- Hierarchical Logic Graphs (HLG) provide a general format for representing both goals and questions that can be grounded by a Soar agent
  - *Based on representation and recursive grounding strategy developed during my thesis for representing and grounding goals, actions, and concept definitions
  - Extending to support questions and new types of tests and calculations

*Kirk, J. (2019). Learning Hierarchical Compositional Task Definitions through Online Situated Interactive Language Instruction (Doctoral dissertation).
Kirk, J. R., & Laird, J. E. (2019). Learning hierarchical symbolic representations to support interactive task learning and knowledge transfer. IJCAI 2019. AAAI Press.

Hierarchical Logic Graphs (HLG)

- HLGs represent general logic in a DAG that the agent can evaluate
  - Represents a conjunction of predicates test (such as apply-property) evaluated bottom up
  - Agent evaluates by performing calculations and grounding to current situation
  - Increases the generality and supported complexity of goals/questions
- Simple example: “What is color of the cup?”
  - Generated DAG:



General Formulation:

$$f(x_1, \ldots, x_m) = \bigcap_{i=1}^{n} f_i(x_j \ldots)\ (1 \geq j \geq m)$$

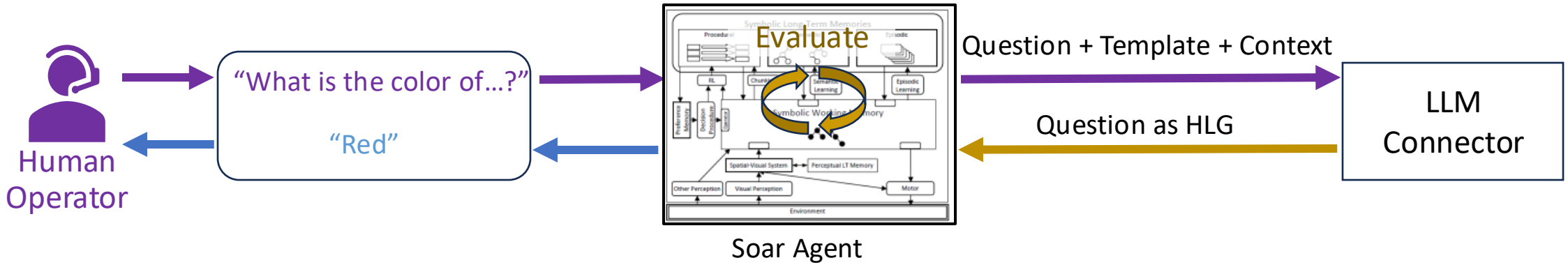
x_j: {objects/entities, values, sets, world states}
f_i: predicate terms
m: number of objects
n: number of terms (f_i predicates)

HLG Representation and Evaluation

- Nodes represent conjunctions of different predicate tests
  - Properties (id, weight, color, cooked-state, dirty-state, speed, …)
  - Math: (max/min, unit conversion, addition, subtraction, division, …)
  - Spatial properties (position…), calculations (distance, …), and relations (on, in, …)
  - Temporal properties (time), relations (while, when, …)
- Agent traverses HLG down and evaluates bottom-up
  - Grounds to the current situation or retrieved situations (episodic memory)
  - Evaluates each predicate using items, sets, and context computed bottom-up from graph's lower layers
  - Produces a successful grounding/answer OR the nodes that cannot be satisfied/grounded

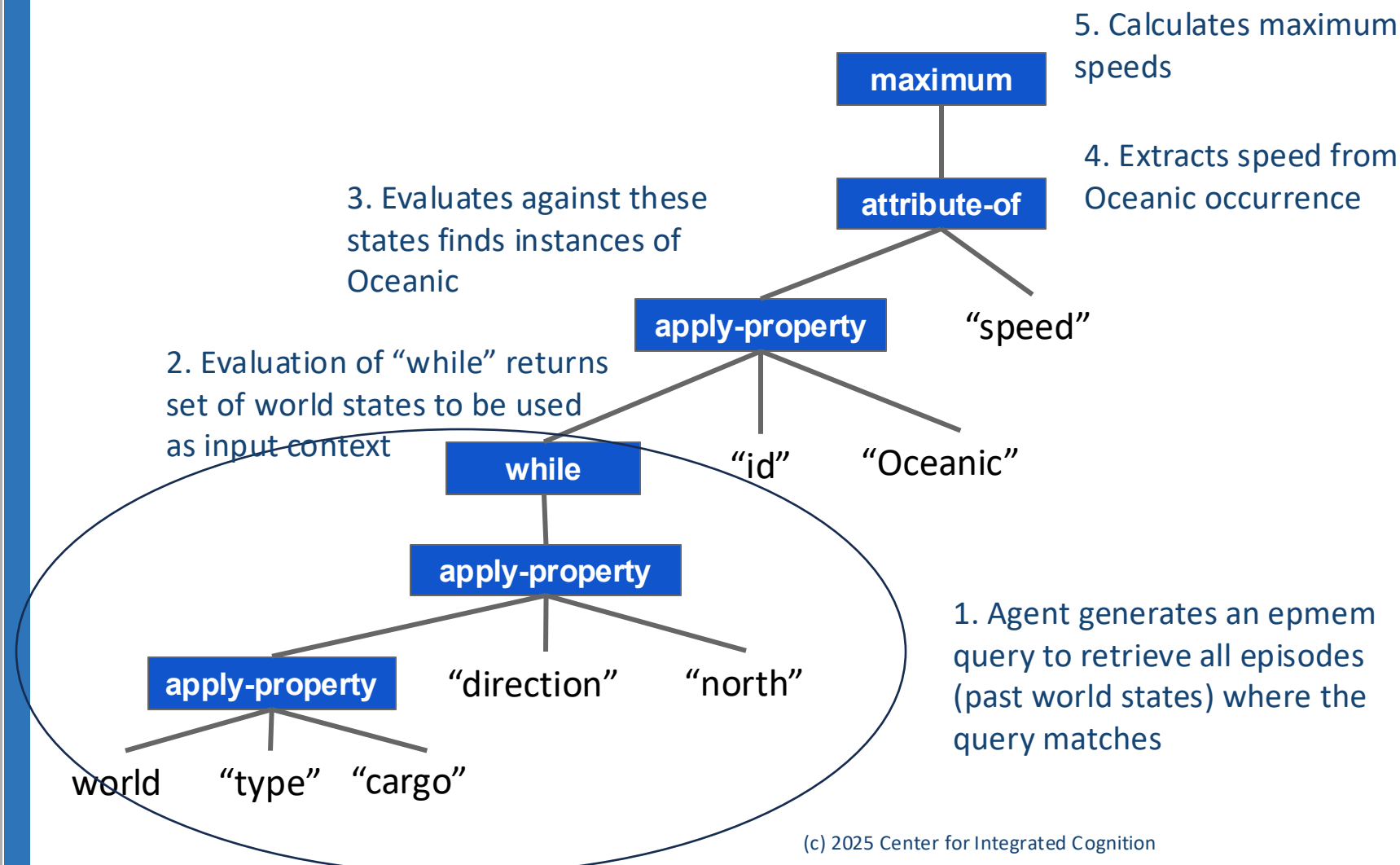
Using LLMs for generation

- Prior work generated HLGs from Soar NLP over limited supported syntax
  - Primarily targets learning goals, concepts, and actions for games & puzzles
- Now via Language Model Connector, agent prompts LLM with user question/goal, agent world context (e.g., objects, properties, relations) , and example conversions to produce a HLG (JSON ↔ Soar)
- Agent then interprets HLG, evaluating nodes and performing required calculations



Historical question example

"While the cargo ship was going north what was the max speed of the Oceanic?"



5. Calculates maximum of those speeds

4. Extracts speed from each Oceanic occurrence

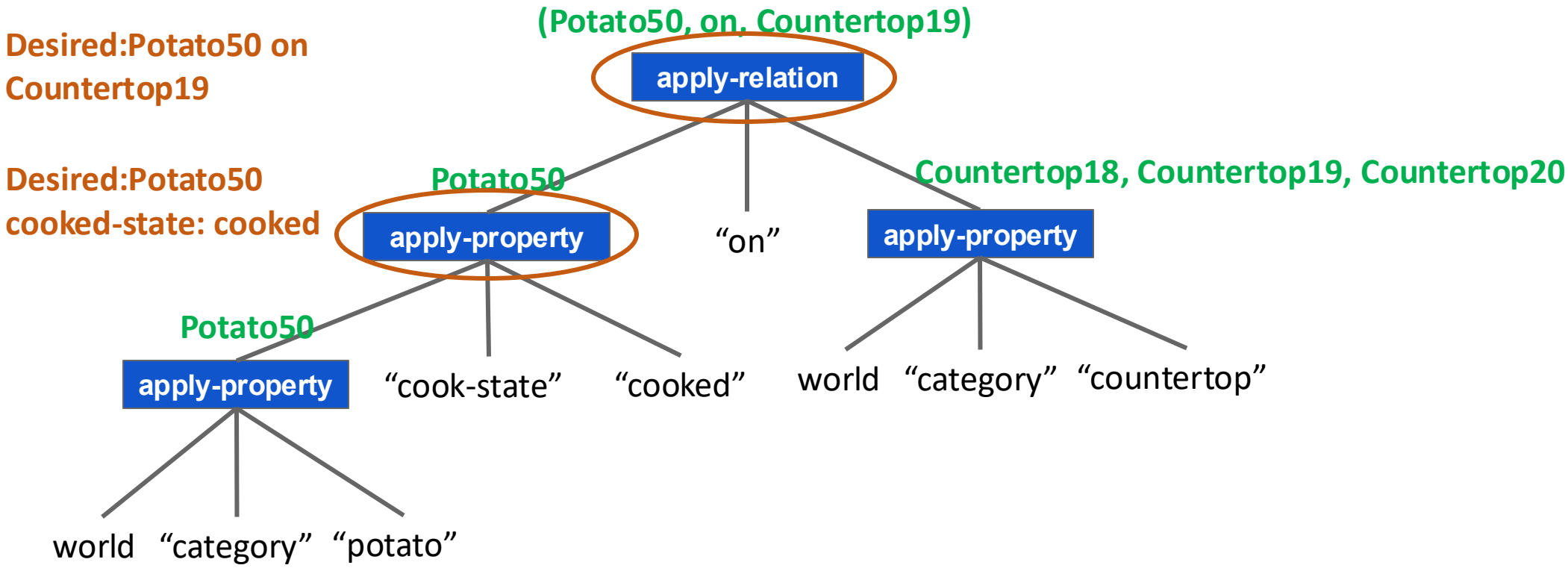
3. Evaluates against these states finds instances of Oceanic

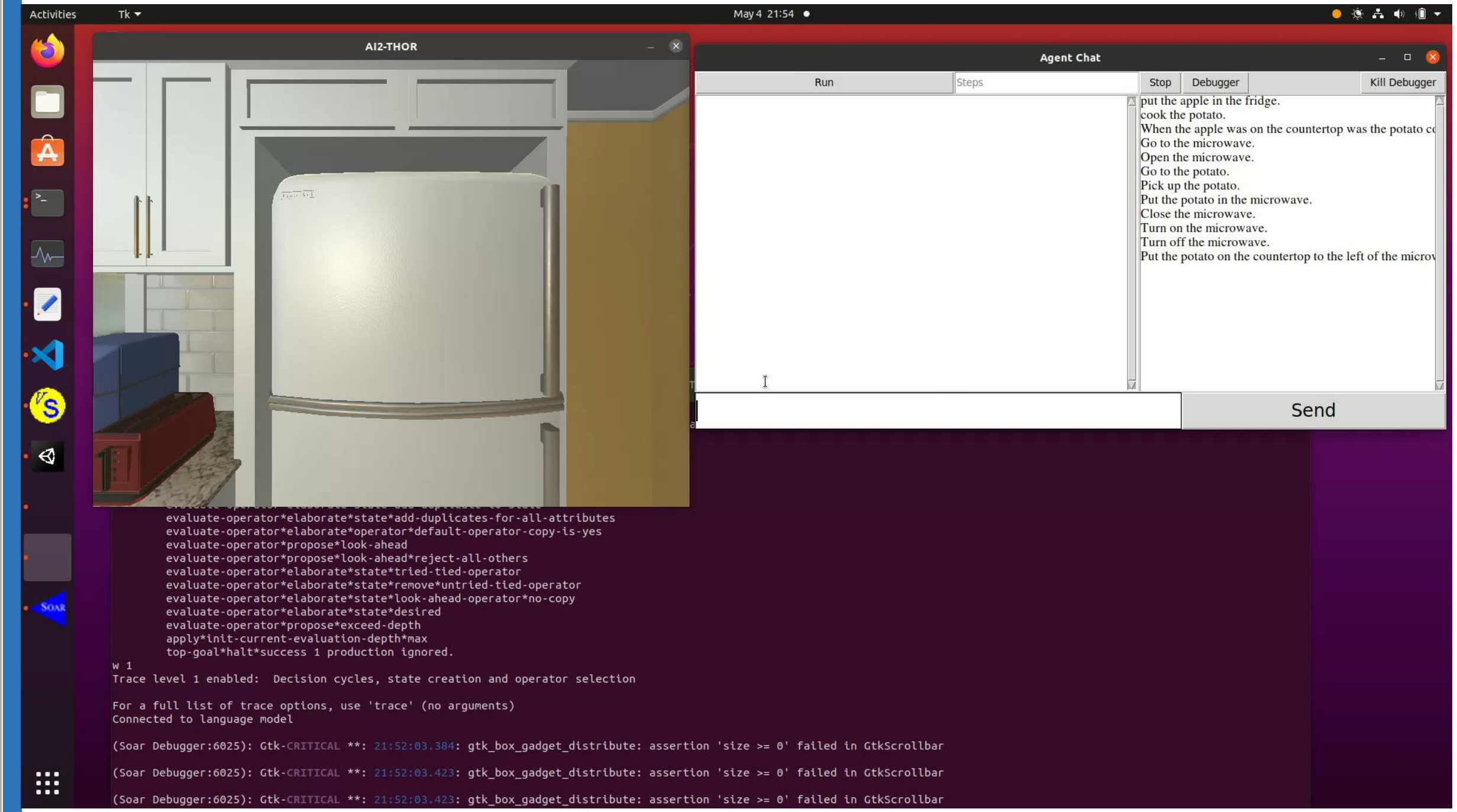
2. Evaluation of "while" returns set of world states to be used as input context

1. Agent generates an epmem query to retrieve all episodes (past world states) where the query matches

Goal example with ThorSoar MEA

"Goal is a cooked potato on the countertop."





QA Comparison of direct LLM vs. HLG

- Initial testing on small set (6) of complex questions
  - More extensive experiments planned
- LLM is given log as context to answer historical questions
- Surprisingly(?) direct LLM answering performs almost as accurately with more advanced models
  - But long response times
  - Expensive LLM usage with long logs
  - Struggles with historical questions when the log is long

Mode (Model)	Number Correct	Average Response Time (seconds)	Rough number of Tokens	$ Cost: 1M Input Tokens
LLM only (4o)	2	0.9	100K	$2.50
LLM only (o3-mini)	5	9.8	100K	$1.10
LLM only (o1)	5-6	18.1	100K	$15.10
HLG (4o)	6	2.4	2.5K	$2.50

Nuggets & Coal

Nuggets

- Keeping thesis work alive!
  - Replacing limited language usage with language interpretation capabilities of LLMs
- Working for multiple purposes (goals and QA)
- Applied in multiple projects/domains
- Showing good integration of CA abilities (episodic memory), LLMs (for NL conversion), and ITL

Coal

- Lacks support currently for complex conjuctions and disjunctions & still some set functionality to port
- Need better strategy for integration of MEA and HLG for more complex goals
- Long prompt due to growing examples
  - Need to experiment with approaches such as finetuning and DSPy
- More experiments to evaluate LLM vs. HLG
- Chunking not implemented yet

Questions?

More examples

- Mathematical Calculations
  - “How many kph is the car going?”
  - Converts internal representation (mph) to kph
- Spatial Properties
  - “What is the distance between the two ships?”
  - Generates and evaluates the *Haversine formula* for computing distance using lat/long
- Conjunctions
  - “At its current speed, how long would it take Ship-A to reach the Ship-B?”
  - Performs calculations, including Haversine formula, unit conversion, and division
- Historical Information
  - “When the Oceanic was leaving Port-A, how many ships were traveling to Port-B?”
  - Retrieves past experiences from *episodic memory*, finds ships traveling to Port-B and counts sets