



**SCHOOL OF SCIENCE & ENGINEERING**

CSC 4301 – 01

**Term Project: Final Report**

**Realized by:**

Adnane Ahroum

Moncef El Kadiri

Reda Elgaf

Zakaria Amrani

**Supervised by:**

Professor RACHIDI, Tajjedine

**FALL 2023**

November 1<sup>st</sup>, 2023

## Table of content:

I. Abstract

II. Introduction

III. Setting up Jess

IV. Project Tasks

- a. Improve the Hunter's reasoning ability.
- b. New Top-level Goal – Killing the Wumpus.
- c. Improve the Hunter's ability to go to a given location
- d. Allow the Hunter to choose the best action among all possible.

V. Conclusion

## **I. Abstract;**

The goal of this Jess-based project is to improve a Hunter's cave exploration abilities. It includes things like improving navigation skills, making better decisions, introducing a new objective of getting rid of the Wumpus, and evaluating overall performance. All these improvements provide the Hunter with greater ability to go through caves more efficiently and strategically.

## **II. Introduction:**

One well-known Artificial Intelligence task that is frequently used as a standard for agent-based reasoning and decision-making is the Wumpus World game. In this case, the agent which we refer to as a knowledge-based agent. The agent's main tasks include exploring a cave, avoiding dangerous pits, avoiding the intimidating Wumpus, finding the gold, and finally making a safe return to the cave's entrance. The goal of this project is to improve the agent's decision-making process by focusing on accuracy and effectiveness in carrying out the necessary tasks. The Wumpus World game has been implemented in a number of programming languages, including Java, Python, and Prolog; however, in this project, we will mainly be using Jess, also referred to as "The Rule Engine for the Java Platform." Jess makes it possible to build rule-based expert systems that perform well with Java, giving our agent's decision-making improvements a strong foundation. The sensors on the agent are the primary information sources and are the core of its decision-making mechanism. The agent can sense stench, breezes, and glitters in the cave thanks to these sensors. A breeze warns of a pit nearby, a stench indicates the existence of a Wumpus nearby, and glittering indicates the presence of gold. As the agent moves through the stages and states of the game, these sensory inputs which are kept in its knowledge base play a crucial role in directing its decisions and movements. In the following sections, we will discuss the tasks and enhancements meant to improve the agent's decision-making skills inside the Wumpus World through Jess, and ultimately optimize its path to accomplish the game's objectives more accurately and efficiently.

### III. Setup:

Before manipulating the agent, we will have to install the Jess engine so that we can run the simulation and manipulate the code as we want:

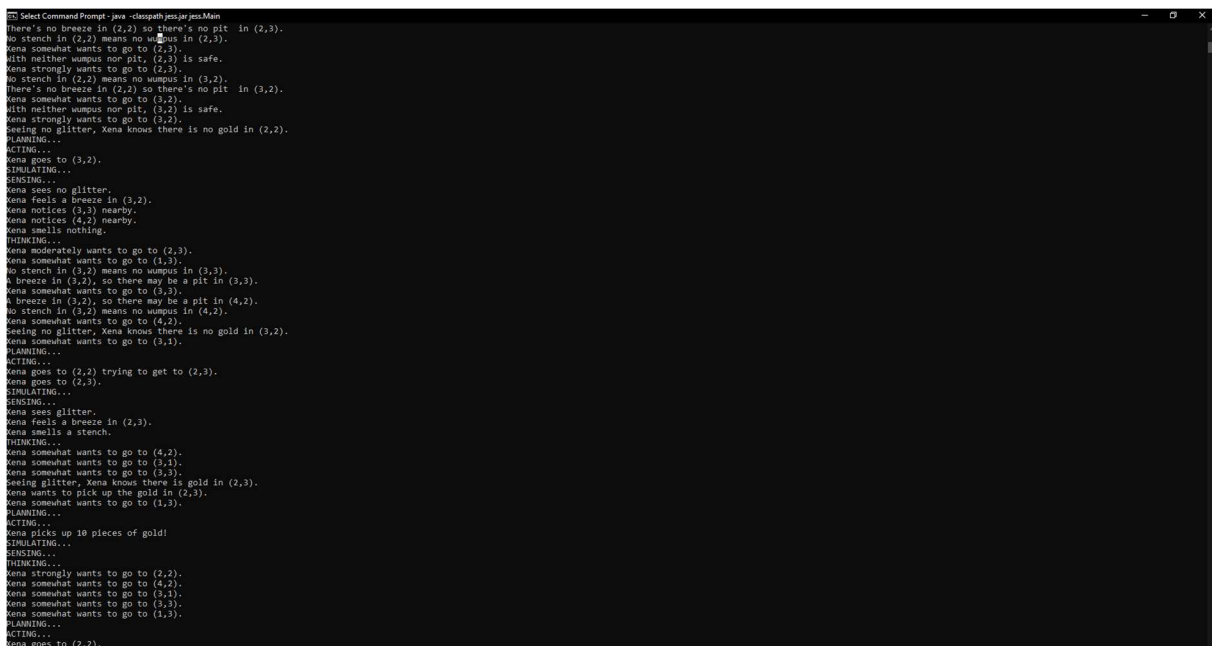
First, we started by downloading the Jess file containing the tools and documentation we need, we can start Jess by using the java command on the terminal to have the following result:

```
MacBook-Air-de-El-4:~ Moncef$ cd ~/Desktop/Project2/JessWumpus/Jess61p4
MacBook-Air-de-El-4:Jess61p4 Moncef$ java -classpath jess.jar jess.Main

Jess, the Java Expert System Shell
Copyright (C) 2001 E.J. Friedman Hill and the Sandia Corporation
Jess Version 6.1p4 7/8/2003

Jess> |
```

Then, we proceed by adding the knowledge base, and the cave files to compile the Jess code and run a simulation of the Wumpus world. After batching the files and running the program, the terminal shows this output:



```
Select Command Prompt - java -classpath jess.jar jess.Main
There's no breeze in (2,2) so there's no pit in (2,3).
No stench in (2,2) means no wumpus in (2,3).
Xena somewhat wants to go to (2,3).
With neither wumpus nor pit, (2,3) is safe.
Xena strongly wants to go to (2,3).
No stench in (2,2) means no wumpus in (3,2).
There's no breeze in (2,2) so there's no pit in (3,2).
Xena somewhat wants to go to (3,2).
With neither wumpus nor pit, (3,2) is safe.
Xena strongly wants to go to (3,2).
Seeing no glitter, Xena knows there is no gold in (2,2).
PLANNING...
ACTING...
Xena goes to (3,2).
SIMULATING...
SENSING...
Xena sees no glitter.
Xena feels a breeze in (3,2).
Xena notices (3,3) nearby.
Xena notices (4,2) nearby.
Xena smells nothing.
THINKING...
Xena moderately wants to go to (2,3).
Xena somewhat wants to go to (1,3).
No stench in (3,2) means no wumpus in (3,3).
A breeze in (3,2), so there may be a pit in (3,3).
Xena somewhat wants to go to (3,3).
A breeze in (3,2), so there may be a pit in (4,2).
No stench in (3,2) means no wumpus in (4,2).
Xena somewhat wants to go to (4,2).
Seeing no glitter, Xena knows there is no gold in (3,2).
Xena somewhat wants to go to (3,3).
PLANNING...
ACTING...
Xena goes to (2,2) trying to get to (2,3).
Xena goes to (2,3).
SIMULATING...
SENSING...
Xena sees glitter.
Xena feels a breeze in (2,3).
Xena smells a stench.
THINKING...
Xena somewhat wants to go to (4,2).
Xena somewhat wants to go to (3,1).
Xena somewhat wants to go to (3,3).
Seeing glitter, Xena knows there is gold in (2,3).
Xena wants to pick up the gold in (2,3).
Xena somewhat wants to go to (1,3).
PLANNING...
ACTING...
Xena picks up 10 pieces of gold!
SIMULATING...
SENSING...
THINKING...
Xena strongly wants to go to (2,2).
Xena somewhat wants to go to (4,2).
Xena somewhat wants to go to (3,1).
Xena somewhat wants to go to (3,3).
Xena somewhat wants to go to (1,3).
PLANNING...
ACTING...
Xena goes to (2,2).
```

This shows that the hunter managed to travel the “cave0” successfully. We will start manipulating the code to perform the required tasks and improve the agent’s algorithm.

## IV. Project Tasks:

### a. Improve the Hunter's reasoning ability:

To complete this game, Hunter's reasoning skills must be strengthened by the addition of rules that allow the system to, whenever doable, find exactly the Wumpus's location using sensory data and logical deduction. The winner's abilities will increase overall, and its odds of winning the game will increase as well thanks to this task.

#### *(i). Code changes:*

- 1- We added a "function" query to return the neighbors that might hold a Wumpus "q-adjacent-that-might-Wumpus."
- 2- We added a "Wumpus-found" rule which determines definitive Wumpus location.
- 3- We did the same thing for the pit and followed the same logic with " q-adjacent-that-might-pit " and "pit-found".

#### *(ii). Execution traces:*

This the execution traces from “cave0” showing the changes:

```
ACTING...
Xena goes to (1,2).
SIMULATING...
SENSING...
Xena sees no glitter.
Xena notices (1,3) nearby.
Xena smells a stench.
Xena notices (2,2) nearby.
Xena feels no breeze in (1,2).
THINKING...
(2,2) is safe, so there's no pit or wumpus in it.
Xena slightly wants to go to (1,2).
Xena strongly wants to go to (2,2).
Xena moderately wants to go to (2,1).
With stench in (1,2), maybe the wumpus is in (1,3).
The wumpus was found in (1,3).
```

```
PLANNING...
ACTING...
Xena goes to (2,3).
SIMULATING...
SENSING...
Xena sees glitter.
Xena feels a breeze in (2,3).
Xena smells a stench.
Xena notices (3,3) nearby.
THINKING...
Xena moderately wants to go to (3,2).
A breeze in (2,3), so there may be a pit in (3,3).
The pit was found in (3,3).
With stench in (2,3), maybe the wumpus is in (3,3).
Seeing glitter, Xena knows there is gold in (2,3).
Xena wants to pick up the gold in (2,3).
Xena somewhat wants to go to (1,3).
PLANNING...
ACTING...
```

#### **b. New Top-Level Goal -- Killing the Wumpus:**

To enhance the Hunter's capabilities in the Wumpus world, we introduce a new top-level goal: killing the Wumpus and finding the gold. This goal requires modifying the existing rule set to incorporate rules that guide the Hunter in eliminating the Wumpus strategically. We also add a "shoot" action and a slot to track the number of arrows available to the Hunter.

This expansion introduces a strategic dimension to the game. The Hunter must now consider proximity to the Wumpus, arrow availability, and its own safety. The modified rule set prioritizes shooting the Wumpus when it's vulnerable and when the Hunter has sufficient arrows. Safety remains a key concern.

The "shoot" action allows the Hunter to actively engage with the Wumpus, and the arrow count ensures strategic arrow usage. These changes empower the Hunter to make more adaptive and effective decisions, increasing its chances of success in the Wumpus world.

(i). Code changes:

1- We updated the hunter's definition to add the arrow slot

```
;;=====Start Change Task 2 =====
(deftemplate hunter "A hunter"

  (slot agent (default Xena))
  (slot x (type INTEGER))
  (slot y (type INTEGER))
  (slot gold (default 0)(type INTEGER))
  (slot alive (default TRUE))
  (slot arrow (default 1)(type INTEGER)))

;;=====End Change Task 2 =====
```

2- We added the action fire an arrow (which also includes a part which handles an error encountered in cave 2 where we had a Wumpus and pit on the same cave)

```
596 ;;=====Start Change Task 2 =====
597
598 (defrule fire-arrow
599   (task act)
600   ?current-goal <- (goal (action shootwumpus))
601   ?hunter-info <- (hunter (agent ?agent)(x ?hx)(y ?hy)(arrow ?arrow-left))
602   ?next-cave <- (cave (x ?nx)(y ?ny)(has-wumpus TRUE))
603   ?current-cave <- (cave (x ?hx)(y ?hy))
604   ?wumpus-info <- (wumpus (x ?wx)(y ?wy) (alive TRUE))
605   ?pit-status <- (cave (x ?nx)(y ?ny)(has-pit ?pit-present))
606   =>
607   (printout t "wumpus located at (" ?wx "," ?wy ")" crlf)
608   (printout t ?agent " fires " ?arrow-left " arrow at the wumpus in (" ?nx "," ?ny ") and eliminates it!" crlf)
609   (modify ?hunter-info (arrow (- 1 ?arrow-left)))
610   (if (eq ?pit-present FALSE) then
611     (modify ?next-cave (has-wumpus FALSE)(safe TRUE))
612   else
613     (modify ?next-cave (has-wumpus FALSE))
614   )
615   (modify ?current-cave (stench false))
616   (modify ?wumpus-info (alive FALSE))
617   (retract ?current-goal)
618 )
619 */
620 ;;=====End Change Task 2 =====
```

3- We added the desire desire-to-shoot-wumpus:

```
446 ;;=====Start Change Task 2 =====
447
448 (defrule desire-to-shoot-wumpus
449   (task think)
450   (hunter (agent ?agent)(x ?hx)(y ?hy))
451   (cave (x ?nx)(y ?ny)(has-wumpus TRUE))
452   (not (desire (agent ?agent) (strength ?*veryhigh*) (action shootwumpus) (x ?nx) (y ?ny)))
453   =>
454   (assert (desire (agent ?agent) (strength ?*veryhigh*) (action shootwumpus) (x ?nx) (y ?ny)))
455   (printout t ?agent " has a strong desire to shoot the wumpus at (" ?nx "," ?ny ")." crlf)
456 )
457
458 ;;=====End Change Task 2 =====
```

4- We changed desire-to-leave-caves to add condition of arrows = 0

```
418 ;;;====Start Change Task 2 =====
419
420 (defrule desire-to-exit
421   (task think)
422   (hunter (agent ?a)(x ?x)(y ?y)(gold ~0)(arrow 0))
423   (cave (x ?x)(y ?y)(has-exit TRUE))
424   =>
425   (printout t "Having found the gold and killed the Wumpus, " ?a " wants to leave the caves." crlf)
426   (assert (desire (agent ?a)(strength ?*veryhigh*)(action leavecaves))))
427
428 ;;;====End Change Task 2 =====
```

(ii). *Execution traces:*

These are the execution traces from cave0 showing the shooting of the wumpus:

```
THINKING...
(2,2) is safe, so there's no pit or wumpus in it.
Xena slightly wants to go to (1,2).
Xena strongly wants to go to (2,2).
Xena moderately wants to go to (2,1).
With stench in (1,2), maybe the wumpus is in (1,3).
The wumpus was found in (1,3).
Xena has a strong desire to shoot the wumpus at (1,3).
There's no breeze in (1,2) so there's no pit in (1,3).
Seeing no glitter, Xena knows there is no gold in (1,2).
Xena slightly wants to go to (1,2).
```

```
ACTING...
Wumpus located at (1,3)
Xena fires 1 arrow at the wumpus in (1,3) and eliminates it!
SIMULATING...
```

### **c. Task 3. Improve the Hunters Ability to Go to a Given Location.**

The goal of this job is to improve the Hunter's navigational skills to a certain place in the cave3.jess environment. To accomplish this, new rules will be added that give priority to newly discovered cave locations and allow the Hunter to investigate more direct routes to the destination. Throughout the analysis of the third task, we can extend our knowledge of the cave with the following observations:

#### **1) Safe:**

- Safe cave slot + adjacent = Very high / high desire.
- Safe cave slot + distant = medium desire.
- Safe cave slot + distant + already visited = low desire.



## 2) Risky:

- Risky cave slot + adjacent = low desire.
- Risky cave slot + distant = very low desire.
- Risky cave slot + distant + already visited = extremely low desire.

### Desire printouts in the output:

- Very high / High desire = “Strongly”
- Medium desire = “Moderately”
- Low = “Somewhat”
- Very low desire = “Slightly”
- Extremely low desire = “Barely”

The inclusion of "Very High" and "High" desires is aimed at prioritizing safety and minimizing risks for the Hunter. Additionally, the introduction of "Extremely Low" desire aims to enhance the searching technique of the agent and maximize the chances of safely navigating and exiting the cave environment.

### (i). Code changes:

1- We added a new global variable for desire strength “extremelylow” with value 0:

```
2  ;;=====Start Change Task 3 =====
3  (defglobal ; these global variables encode the strength of desires
4    ?*veryhigh* = 5
5    ?*high* = 4
6    ?*medium* = 3
7    ?*low* = 2
8    ?*verylow* = 1
9    ?*extremelylow* = 0)
10 ;;=====End Change Task 3 =====
11
```

2- We added three new rules:

The rules in the code are focused on adding desires related to moving to different caves, depending on their safety and distance from the current position of the agent. The desires are added with different strengths, ranging from very low to extremely low, depending on the level of safety and distance of the cave.

By adding desires with different strengths, the agent can prioritize actions that align with its goals. For example, if the agent's goal is to find the gold in a safe and efficient way, desires with higher strengths, such as going to a safe and distant cave, will be promoted to goals before desires with lower strengths, such as going to a risky and adjacent cave. This helps to improve the efficiency of the hunter code by guiding the agent towards actions that are more likely to satisfy its goals, while

avoiding actions that are less likely to do so. This reduces the time and resources required to achieve the agent's objectives and increases the likelihood of success.

```
509 ;;====Start Change Task 3====
510
511 (defrule add-desire-to-go-to-visited-safe-distant-cave
512 "go to a non-adjacent, safe, visited cave"
513 (task think)
514 (hunter (agent ?agent)(x ?x)(y ?y))
515 (cave (x ?x2)(y ?y2)(visited TRUE)(safe TRUE))
516 (not (adj ?x ?y ?x2 ?y2))
517 =>
518 (printout t ?agent " slightly wants to go to (" ?x2 ", " ?y2 ")." crlf)
519 (assert (desire (agent ?agent) (strength ?*verylow*) (action go)(x ?x2)(y ?y2))))
520
521
522 Code Suggestions
523
524 (defrule add-desire-to-go-to-visited-risky-adjacent-cave
525 "go to an adjacent, risky, visited cave"
526 (task think)
527 (hunter (agent ?agent)(x ?x)(y ?y))
528 (cave (x ?x2)(y ?y2)(visited TRUE)(safe UNKNOWN))
529 (adj ?x ?y ?x2 ?y2)
530 =>
531 (printout t ?agent " barely wants to go to (" ?x2 ", " ?y2 ")." crlf)
532 (assert (desire (agent ?agent) (strength ?*extremelylow*) (action go)(x ?x2)(y ?y2))))
533
534 (defrule add-desire-to-go-to-visited-risky-distant-cave
535 "go to a distant, risky, visited cave"
536 (task think)
537 (hunter (agent ?agent)(x ?x)(y ?y))
538 (cave (x ?x2)(y ?y2)(visited TRUE)(safe UNKNOWN))
539 (not (adj ?x ?y ?x2 ?y2))
540 =>
541 (printout t ?agent " barely wants to go to (" ?x2 ", " ?y2 ")." crlf)
542 (assert (desire (agent ?agent) (strength ?*extremelylow*) (action go)(x ?x2)(y ?y2))))
543 */
544 ;;====End Change Task 3====
```

## (ii). Execution traces:

These are the execution traces from cave3 showing the agent successfully exiting the cave after accomplishing all goals:

```
ACTING...
Seeker goes to (1,3).
SIMULATING...
SENSING...
Seeker sees no glitter.
Seeker notices (1,4) nearby.
Seeker feels no breeze in (1,3).
Seeker smells nothing.
THINKING...
(1,4) is safe, so there's no pit or wumpus in it.
Seeker slightly wants to go to (1,3).
Seeker strongly wants to go to (1,4).
Seeker slightly wants to go to (1,1).
Seeker moderately wants to go to (2,1).
Seeing no glitter, Seeker knows there is no gold in (1,3).
Seeker slightly wants to go to (1,3).
```

#### **d. Task4. Extra Credit: Allow the Hunter to choose the best action among all possible**

This task involves allowing the Hunter to choose the best action among all possible. To accomplish this, we added safe-cave3 and x-unsafe move rules to ensure the Hunter only moves to safe caves and avoids dangerous ones. We also changed the choose-desire rule to print the chosen action, enabling the Hunter to make more informed decisions and increase its chances of success. By implementing these features, the Hunter will be able to navigate the cave more effectively and make optimal use of its resources.

##### *(i). Code changes:*

- 1- We changed safe-cave3: Identifies safe cave locations, sets "has-wumpus" and "has-pit" to FALSE, and "safe" to TRUE.
- 2- We added X-unsafe move: Blocks unsafe moves due to pits, asserts "task think" for the hunter to reconsider.
- 3- We added Wumpus-location
- 4-We added pit-location
- 5-Adding global variable for desires extremely low that has the value 0

```

376 ;;====Start Change Task 4 =====
377 (defrule pit-location
378
379   "Mark the pit's location"
380   (task think)
381   ?f <- (cave (x ?x) (y ?y) (safe UNKNOWN))
382   (pit (x ?x) (y ?y))
383   =>
384   (modify ?f (has-pit TRUE)))
385
386 (defrule wumpus-location
387   "Mark the wumpus's location"
388   (task think)
389   ?f <- (cave (x ?x) (y ?y) (safe UNKNOWN))
390   (wumpus (x ?x) (y ?y))
391   =>
392   (modify ?f (has-wumpus TRUE)))
393   */
394 (defrule safe-cave3
395   "safe => ~wumpus ^ ~pit"
396   (task think)
397   ?f <- (cave (x ?x) (y ?y) (safe UNKNOWN))
398   (not (wumpus (x ?x) (y ?y)))
399   (not (pit (x ?x) (y ?y)))
400   =>
401   (printout t "(" ?x " ", " ?y ") is safe, so there's no pit or wumpus in it." crlf)
402   (modify ?f (has-wumpus FALSE) (has-pit FALSE) (safe TRUE)))
403
404 (defrule x-unsafe-move
405   (declare (salience 100))
406   (task move)
407   (hunter (agent ?agent) (x ?x) (y ?y) (alive TRUE))
408   (cave (x ?new-x) (y ?new-y) (safe ~TRUE) (has-pit TRUE))
409   (test (xy-next ?x ?y ?new-x ?new-y))
410   =>
411   (printout t ?agent " will not move to (" ?new-x " ", " ?new-y ") because it's unsafe." crlf)
412   (assert (task think)))

```

```

553 (defrule choose-desire
554   "pick the best desire available for a given action. note that we
555   will only promote one desire to be a goal at a time."
556   (task plan)
557   ?f1 <- (desire (strength ?s1)(action ?act1)(x ?x1)(y ?y1))
558   (not (desire (strength ?s2:> ?s2 ?s1))))
559   ?f2 <- (desire (strength ?s3:< ?s3 ?s1)(action ?act2)(x ?x2)(y ?y2))
560   (not (goal))
561   =>
562   (printout t "Choosing move with priority " ?s1 " (action: " ?act1 " , x: " ?x1 " , y: " ?y1 ") over move with priority " ?s3 " (action: " ?
563   (retract ?f1)
564   (assert (goal (action ?act1) (x ?x1)(y ?y1))))
565
566 ;;====End Change Task 4 =====
567
568 14 ACT rules

```

(ii). *Execution traces:*

For this one, we are printing the score after running cave3, before and after making the changes, along with the printed change 5 which shows the prioritization taking place.

```
ACTING...
Xena goes to (1,3).
SIMULATING...
SENSING...
THINKING...
Xena strongly wants to go to (1,2).
Xena slightly wants to go to (1,3).
Xena moderately wants to go to (2,2).
Xena slightly wants to go to (1,1).
Xena moderately wants to go to (2,1).
PLANNING...
Xena chooses move with priority 5 (x: 1, y: 2) over move with priority 1 (x: 1, y: 3).
ACTING...
Xena goes to (1,2).
SIMULATING...
SENSING...
THINKING...
Xena slightly wants to go to (2,3).
Xena slightly wants to go to (1,4).
Xena strongly wants to go to (1,1).
Xena slightly wants to go to (1,2).
Xena moderately wants to go to (2,1).
Xena strongly wants to go to (2,2).
PLANNING...
Xena chooses move with priority 5 (x: 1, y: 1) over move with priority 1 (x: 1, y: 2).
ACTING...
Xena goes to (1,1).
Xena leaves the caves.
187
Jess>
```

## V. Conclusion:

In conclusion, this project provided an intriguing approach to studying the behavior of a reflex agent in the well-known Wumpus World game using the Jess rule engine. The utilization of Jess introduced a distinct coding experience compared to traditional programming languages. The project not only enhanced our understanding of representation and inferencing techniques for problem-solving but also deepened our expertise in utilizing first-order logic to code in Jess and analyze the agent's output from various perspectives. Through this project, we gained a sense of fulfillment by witnessing the manifestation of artificial intelligence in our agent, which demonstrated the ability to make intelligent decisions based on incomplete or uncertain knowledge. Initially, we encountered challenges in working with Jess, as it was an entirely new tool for us. It required time and effort from the team members to comprehend the documentation and effectively utilize its capabilities. Additionally, the limited project timeline posed constraints, making it difficult to address and resolve errors generated by the Jess engine in a timely manner. However, one of the most rewarding aspects of the project was the collaborative process of tracing the agent's path on paper, drawing the grid, and applying logical reasoning to validate our code changes. This exercise fostered a deep understanding of the agent's behavior and facilitated effective debugging.

Overall, this project provided valuable insights and equipped us with a range of techniques that will undoubtedly prove beneficial when tackling modern and complex AI problems in the future.