**CSCI 446 — Artificial Intelligence**

**Project #2**
**Logical Inference and the Wumpus World**

The purpose of this assignment is to give you a chance to implement a basic, first-order logic reasoning system. In this project, you will implement a system that does unification and resolution on clauses. We will be assuming that the rules are already encoded in clause form. You will be testing your system on various instances of the Wumpus world.

For this project, you need to write a "problem generator" whereby you will create several Wumpus worlds of various sizes. The Wumpus world cave will be generated as a square consisting of several cells. You will assume that each cell in the Wumpus world is either empty, contains gold, contains a wumpus, is blocked, or is a bottomless pit. As input, you will provide a probability of generating a pit $P_{\text{pit}}$, a probability of generating an obstacle $P_{\text{obs}}$, and a probability of generating a wumpus $P_{\text{wumpus}}$.

After generating the world, you will select one of the empty cells at random (i.e., using a uniform probability distribution over the empty cells) and place the gold there. The initial state of the game will be selected, also at random, from one of the remaining empty cells. Your explorer will have one arrow for every monster you generate, so you may assume the cave has been scouted previously to determine how many monsters were there. Once the gold is found, you may assume the agent teleports to safety, so treat the gold cell as the goal state.

You also need to generate first-order rules (not propositional!) to tell the explorer how to behave. As described in class, the explorer can only see the current cell, can smell adjacent cells, and can feel wind from agent cells. It can also hear a scream from anywhere in the cave, so if an arrow finds its mark, the explorer will know. These rules may be provided in clause form, so you do not need to create a rule converter.

After building the Wumpus worlds, you will use your rule set with your reasoning system and have the explorer explore each cave. You should keep track of the following statistics: Number of times the gold is found, number of wumpus killed, number of times the explorer falls in a pit, number of times the wumpus kills the explorer, total number of cells explored. You should also create a simple reactive explorer that does not use the reasoning system. Instead, it makes a decision which cell to enter at random based on whether or not it believes the neighboring cell is safe. So it selects first from safe neighboring cells, next from unsafe neighboring cells. Record the same statistics for this reactive explorer.

Here are the specific steps that need to be followed:

- Implement the Wumpus world generator.
- Generate at least 10 different Wumpus worlds of sizes ranging from $5 \times 5$ to $25 \times 25$ in steps of 5 (i.e., $5 \times 5$, $10 \times 10$, $15 \times 15$, $20 \times 20$, $25 \times 25$).
- Implement a resolution-based inference system that will analyze each state the explorer is in, update the explorer's knowledge of the world, and use that knowledge to have the explorer make a decision. You should keep track of known safe cells as well as a "frontier" of cells for exploration and use those to guide the decision-making process.
- Write a design document that outlines the architecture of your software (a UML class diagram is encouraged but not required), design decisions made in implementing your algorithms, and the experimental design for testing the results.
- Run experiments, keeping track of the main decisions being made for each algorithm. The count of these decisions will be used to gauge run time since CPU time and wall clock time are not reliable estimators. Make sure you run at least 10 experiments for each cave size (preferably more) to get reasonable statistics.
- Write a paper that incorporates the following elements, summarizing the results of your experiments. Make sure you explain the experimental setup, the tuning process, and the final parameters used for each algorithm.

    1. Title and author names

2. A brief, one paragraph abstract summarizing the results of the experiments
3. Problem statement, including hypothesis (how do you expect the algorithms to do?)
4. Description of algorithms implemented
5. Description of your experimental approach
6. Presentation of the results of your experiments
7. A discussion of the behavior of your algorithms, combined with any conclusions you can draw
8. Summary
9. References (you should have at least one reference related to each of the algorithms implemented, a reference to the data sources, and any other references you consider to be relevant)

- Create a video that is no longer than 5 minutes long demonstrating the functioning of your code. This video should focus on behavior and not on walking through the code. You need to show input, data structure, and output. How you do this is entirely up to you, but be sure it will convince the grader that your program works.
- Submit your fully documented code for the data converter, the video demonstrating the proper functioning of each algorithm, your design document, and your paper (which will include the results of the experiments).

**Due Dates:**

- Design document – October 4, 2021
- Fully documented program code – October 15, 2021
- Video demonstrating code functionality – October 15, 2021
- Project report – October 15, 2021