Design Document: First-Order Logic Reasoning System

Taron Moe-Stull
taronmoe@icloud.com

September 2025

System Overview

This document describes the design of a First-Order Logic (FOL) reasoning system capable of evaluating safety in grid-based environments inspired by the Wumpus World domain. The system ingests structured cave descriptions, constructs a knowledge base of logical facts and rules, and applies unification and resolution to determine whether a target cell is safe, unsafe, or risky.

The design emphasizes transparency, correctness, and traceability. All logical clauses involved in reaching a conclusion are recorded and exported alongside the final determination, enabling both verification and interpretability.

System Requirements

The system parses structured text input describing a cave environment, including grid size, the number of hazards, a traversal path with perceptual information (breezes and stenches), and a query coordinate whose safety must be evaluated.

Each query cell is classified as:

- **Safe**: no possible configuration leads to death
- **Unsafe**: death is guaranteed
- **Risky**: death is possible but not certain

If neither safety nor unsafety can be proven, the cell is labeled risky by default.

Input Format

Input is provided via a plain text file containing:

- Grid dimensions
- Number of arrows (Wumpus instances)
- A traversal path with perceptual data (Breeze: T/F, Stench: T/F)
- A query coordinate
- A reference resolution used to validate correctness

Output Format

For each evaluated puzzle, the system produces a text file listing:

- All base and derived clauses used during inference
- The final determination in the form:

```
QUERY: SAFE | UNSAFE | RISKY
```

Output files follow a consistent naming scheme for traceability.

Reasoning Model

The system implements general First-Order Logic reasoning using:

- Constants, variables, and predicates
- Unification to bind variables to terms
- Resolution to derive new clauses from existing knowledge

Rules are expressed in conjunctive normal form and applied iteratively to expand the knowledge base. Resolution continues until either safety or unsafety is proven, or no new information can be derived.

This approach generalizes across both Wumpus World scenarios and a smaller toy knowledge base used for correctness validation.

System Architecture

The system is organized around a central knowledge base that accumulates both observed facts and inferred clauses derived through logical reasoning. Input is provided via structured cave path text files, which describe grid dimensions, hazard counts, traversal paths, perceptual observations, and a query cell whose safety must be evaluated.

Parsed input facts are injected into the knowledge base alongside a fixed set of Wumpus World rules. These rules encode the semantic relationships between percepts and hazards, such as the implication of a breeze indicating an adjacent pit or a stench indicating an adjacent Wumpus.

Logical inference is performed through the interaction of unification and resolution. Unification matches variables and terms across clauses, producing substitutions, while resolution combines compatible clauses to derive new logical consequences. Newly derived clauses are fed back into the knowledge base, allowing the system to incrementally expand its understanding of the cave.

A query engine operates over the evolving knowledge base to determine whether the target cell can be proven safe or unsafe. If neither can be conclusively derived, the cell is classified as risky. Throughout execution, the system tracks metrics related to unification and resolution activity, and records all clauses involved in the final determination.

The system is designed to run within a notebook environment, where a simple interface allows selection of puzzle files and displays results, metrics, and final classifications. Output artifacts include a detailed log of reasoning steps and a terminal query classification.

All components are designed to be modular and independently testable.

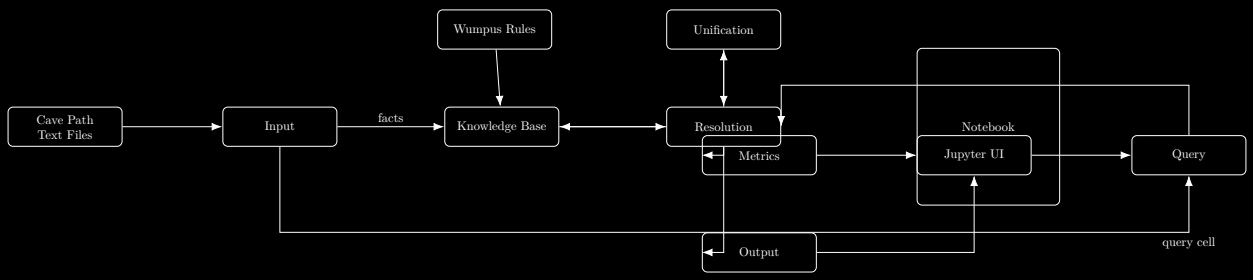

Figure 1: Block diagram of the First-Order Logic reasoning system and data flow.

System Flow

Execution begins by loading a cave description file and parsing its contents into structured facts representing grid geometry, percepts, and traversal information. These facts are inserted into the knowledge base alongside a fixed inventory of Wumpus World rules.

Once initialized, the system constructs logical statements corresponding to the query cell and attempts to reason about its safety. This process alternates between unification and resolution, deriving new clauses and reinserting them into the knowledge base as inference progresses.

If a contradiction is found when assuming the cell is unsafe, the cell is classified as safe. Conversely, if assuming safety leads to contradiction, the cell is classified as unsafe. If neither assumption can be disproven after exhausting all possible resolutions, the cell is labeled risky.

Throughout execution, the system logs every clause involved in the final determination, tracks inference metrics, and produces a structured output artifact that documents both the conclusion and the reasoning path that led to it.

Metrics and Validation

The system tracks:

- Number of variable unifications
- Number of resolution steps
- Final classification accuracy compared to reference answers

These metrics are attached to each output file and serve both as performance indicators and as debugging aids during development.

Testing Strategy

Testing is performed incrementally, beginning with validation of the input parser to ensure correct interpretation of grid structure and perceptual data. Intermediate reasoning steps are verified using controlled scenarios and isolated logic tests before full system integration.

Logical operations such as unification and resolution are tested independently to ensure correctness prior to being composed into the full reasoning pipeline. Additional validation is performed by comparing system output against known solutions provided with the input puzzles.

Implementation Notes

The system is designed to run within a notebook environment, allowing interactive selection of puzzle files and inspection of metrics. All reasoning steps are logged explicitly, prioritizing clarity and debuggability over opaque optimization.

Care is taken to ensure the system remains robust even when portions of the input are missing or underspecified, such as environments with fewer hazards.

References

1. Russell, S., & Norvig, P. *Artificial Intelligence: A Modern Approach*, Chapters 7–8.
2. Barbosa, H. Unification and Resolution.
3. GeeksforGeeks. Resolution Algorithm in Artificial Intelligence.
4. Tutorialspoint. Unification in First-Order Logic.
5. Mermaid Live Editor. Diagramming and flow visualization.