# Assessment 5: Logic-based Agents

**Step-by-Step Instructions with Comments in Python Code**

# Importing required libraries

from pysat.solvers import Glucose3

# Step 1: Define the grid size and initialize the environment

GRID\_SIZE = 4 # 4x4 grid

# Define the knowledge base as a list of clauses

knowledge\_base = []

# Add known facts (example: Breeze in cell (1, 2), Stench in cell (2, 1))

knowledge\_base.append(("Breeze", 1, 2))

knowledge\_base.append(("Stench", 2, 1))

# Step 2: Define rules for inference

# For example, if there's a Breeze, there's a Pit in an adjacent cell

def add\_breeze\_rule(kb, x, y):

adjacent\_cells = [

(x - 1, y), (x + 1, y), (x, y - 1), (x, y + 1)

]

for cell in adjacent\_cells:

if 1 <= cell[0] <= GRID\_SIZE and 1 <= cell[1] <= GRID\_SIZE:

kb.append(("Pit", \*cell))

# Add rules to the knowledge base

add\_breeze\_rule(knowledge\_base, 1, 2)

# Step 3: Implement the propositional logic inference mechanism

def infer(knowledge\_base, query):

solver = Glucose3()

for clause in knowledge\_base:

if clause[0] == "Breeze":

solver.add\_clause([-1]) # Example of encoding a clause

elif clause[0] == "Stench":

solver.add\_clause([2]) # Example of encoding another clause

return solver.solve(assumptions=[query])

# Step 4: Test the agent in the Wumpus World environment

def test\_agent():

# Initial position

position = (1, 1)

while True:

print(f"Agent at {position}")

# Check for safety

if infer(knowledge\_base, ("Safe", \*position)):

print(f"{position} is safe.")

else:

print(f"{position} is not safe.")

break

# Run the test

test\_agent()