

The purpose of this problem set is to gain experience with logic-based methods. In the engineering design section you will be simulating the wumpus world. You will then design and implement a knowledge-based agent to navigate the environment.

You can complete the exercises by either directly marking up this pdf, or by printing, completing, and scanning as a pdf. You should complete the Engineering Design Problems by writing the python code as instructed. The resulting pdf and python files should be uploaded to Canvas via the assignment tab by the due date and time.

## Exercises

1. Given two propositions  $A$  and  $B$ , show that  $A \Leftrightarrow B$  is logically equivalent to  $A \vee B \Rightarrow (A \wedge B)$  using
  - (a) (3 points) truth tables

- (b) (3 points) substitutions using the basic logical identities

2. Given the following PL KB

$$(A \Rightarrow B) \wedge (B \wedge D \Rightarrow C) \wedge (C \Rightarrow E) \wedge (A \wedge D)$$

Prove  $E$  or show it cannot be proven using

(a) (5 points) modus ponens

(b) (5 points) resolution



4. Given the following arbitrary FOL sentences,  $S_1$  and  $S_2$ , determine the MGU if one exists. Variables are lowercase, constants uppercase, functions are denoted by  $()$ .

(a) (2 points)  $S_1 = g(f(A, B), A)$  and  $S_2 = g(x, y)$

(b) (2 points)  $S_1 = f(g(A, y))$  and  $S_2 = f(g(B, A))$

(c) (2 points)  $S_1 = g(A, f(x), f(y))$  and  $S_2 = g(x, f(B), y)$

(d) (2 points)  $S_1 = g(A, f(B, C))$  and  $S_2 = g(x)$

(e) (2 points)  $S_1 = f(g(x), h(g(A), B, y))$  and  $S_2 = f(x, h(x, y, C))$

5. Given the following FOL KB

$$(P1(x) \wedge P2(x) \rightarrow P3(x)) \wedge (P3(x) \wedge P4(x) \rightarrow P5(x))$$

(a) (2 points) Suppose the agent is told  $P1(A) \wedge P2(A)$ , use forward chaining to update the KB

(b) (2 points) Suppose now the agent is told  $P5(A)$ , using backward chaining what is the result of the query  $P4(x)$ ?

6. You have likely encountered the Tower of Hanoi problem before. There are three rods and a number of disks of different sizes which can slide onto any rod. The problem begins with a stack of disks in ascending order of size on one rod. The goal is to move the entire stack to another rod according to the following rules:

- only one disk can be moved at a time.
  - only the top-most disk in a stack can be moved
  - disks must always be stacked in ascending order according to size
- (a) (2 points) Define a KB that can represent this problem for arbitrary number of disks

(b) (2 points) To use your KB above to solve a specific problem instance what additional KB entries are needed?

## Engineering Design Problems

The purpose of this problem set is to gain experience with first-order logic and knowledge-based agents. The engineering design problem is specified more loosely than in previous assignments and so this assignment has a larger program design element and some room for creativity. Note, there is also a documentation requirement. You can download starter code in the `ps2.zip` file from the website.

The basis of this assignment is the wumpus world as defined in section 7.2 of the AIAMA text. You will design and implement both a simulator for this world and an agent to navigate it. You have considerable freedom with respect to design and implementation details, subject to the following functional specifications. The most important of these is that the simulator and agent can **only** communicate through calls to update the agent's percept and to get the agent's action. There can be no other communication between the modules, in particular no "peeking" from the agent into the simulation module.

You should upload the code and any supporting files as a zip file attached to the Assignment on Canvas.

### 7. (10 points) Wumpus World Simulator

In a python file called `wwsim.py` implement a system to populate valid random wumpus worlds and simulate an agent's interaction. Your code should be able to represent a given wumpus world, determine a correct percept sequence given the current state, and simulate the result of an agents action.

Provide a graphical user interface (GUI) using Tkinter to show the current state of the simulated environment, including the relevant status of pits, wumpus, gold, and agent in iconic form. Provide buttons to start a new simulation and make one percept-action time-step. Your GUI should show the current percept sequence, the last action emitted by the agent, and the current performance score. Focus on basic functionality and good user interface design rather than appearance (i.e. ugly glyphs are fine as long as they clearly indicate status).

Your file should import the `wwagent` module described below. The basic loop of your program should be to obtain the current percept sequence and pass it to the agent as a tuple or list in a function call. Then call the agent action function to get the next action. See the agent definition below. Update the performance score and simulate the agent given the returned action, providing an updated percept. This should continue until the agent dies, climbs from the pit (with or without the gold), or a new game is started.

Your program should also be able to run in a non-gui unattended mode, writing a human readable text file describing the simulation to standard output. This mode should be selectable as a command-line flag.

### 8. (10 points) Wumpus World Agent

In a python file called `wwagent.py` implement an agent to navigate the simulated wumpus world. The agent should use a knowledge-base to represent the rules of the environment

and the status of its current model. Inference should be used in conjunction with state-space search to choose a rational action. Your agent can use memory external to the knowledge base.

Your agent module should provide two functions, `update` taking a single tuple, named tuple, or list of percepts and `action` taking no arguments and returning a single action. No other communication between the modules is allowed.

You can (but do not have to) use code from the book, either directly, or for ideas. I have provided the relevant files in the starter code.

9. (10 points) Documentation

You are required to fully document your program at the module, class, and method/function level. You should also include a narrative explaining your design choices, how your code works, and typical results you get when using it. In particular how rational is your agent. Describe any weaknesses or improvements you might make given more time.

How you do your documentation is up to you, however I suggest that you look into the Sphinx document generator as it was designed for literate programming with python. See <http://sphinx-doc.org/> for details.

The final documentation file format should be pdf.