School of Electrical Engineering and Computer Science, Washington State University
Fall 2016

CptS 440/540 Artificial Intelligence

Homework 4

Due: Friday, December 2, 2016 (before midnight / 11:59pm)

General Instructions: Put your answers to the following problems into electronic form. Please submit your entire *non-coding part* of HW4 as a single PDF document by the above deadline. For the coding portion, your answer must contain the following: Agent.h and Agent.cc files, and the mandatory README.txt file. The components of the submission for the programming exercise are the same as the components for the coding part of HW3.
Please collect all these homework files into one zip file and submit as an attachment under **Assignments → Homework 4** for your course (either Pullman 440/540 or Tri-Cities 440/540) on the BB-Learn by the above deadline.

Please submit this assignment, and the remaining HW assignment through the rest of this semester, only once. [While I will grant exceptions if really necessary (e.g., technical glitches or such), I also want to minimize the burden on the TA, as well as minimize the grading turn-around time to the extent possible, given the size of this class and that we have only one TA.]

This assignment includes relatively modest hands-on coding components. Please include the code and supplementary files (incl. a screenshot and scores/results of your test-cases – see below for details) as parts of your Programming Exercise answer (Problem 1 below). The TA may contact you after you have submitted to ensure that your code runs as claimed in your report.
Please write your own code / do not plagiarize other students' code or code found online.

Note: For Pullman students, paper hard copy submission (for the non-programming exercises) remains acceptable, actually encouraged. However, ***please also make sure to upload your HW assignment online***, so that we don't run into problems with BB-Learn and in particular entering your score there later.

1. (Programming exercise – 50 points total)

A C++ Wumpus World simulator, following the rules from the textbook, is available at prof. Holder's teaching website, http://www.eecs.wsu.edu/~holder/courses/AI/wumpus (the current version is 2.7). You will modify/refine your Agent from HW3; in particular, your modified Agent is tasked to solve the following problem: grabs the gold (if possible; see below) and climb out of the cave for any Wumpus World satisfying the following conditions:

a. The world is a 4x4 square grid.

b. The gold is in location (4,4), but your Agent does not know that / cannot use that information.
c. Assume the initial Agent's location is (1,1). Specify your assumption on initial direction of where is the agent looking. (Remember: the Agent can move, or shoot its arrow, only up/down and left/right; i.e., no diagonal moves are possible.)
d. The Wumpus can be anywhere in the grid, other than locations (1,1) or (4,4).
e. Assume, there is between 1 and 3 pits. The pit(s) can be anywhere, other than (1,1). In particular, a pit may be co-located with the gold.
   State clearly in your README file what assumptions you are making w.r.t. existence and position of pits.

Your goal is to design an agent for this Wumpus World with the properties (you are welcome to add additional capabilities, but not required to beyond the specified 'agent capabilities').

a. The agent should always either succeed in *Grab(ing)* the gold before leaving the cave, or else determine that this cannot be done safely, generate an output statement to that effect (i.e., print a line such as "I have determined, I cannot safely snatch the gold!"), and return to (1,1) and get out of the cave alive without the gold.
b. The agent should always *Climb* when in location (1,1) and has already (previously) either grabbed the gold, or determined gold cannot be reached safely.
c. The agent should keep track of safe locations, i.e., locations you are sure do not contain a live Wumpus or a pit.
d. The Agent should *Shoot* the arrow only once it is certain of the Wumpus's location. (In particular, *do not shoot the arrow as soon as stench is detected like we did in the previous assignment.*) If the agent kills the Wumpus, then stenches should be ignored when determining safe locations.

Begin by reading the `readme.txt` file that comes with the simulator. You will be modifying the `Agent.h` and `Agent.cc` files to implement your agent. You should also submit a `readme.txt` file containing any information you think we may need to know about your agent and implementation. Your agent shouldn't require any user input.

You should test your agent on at least 3 initial configurations (with different choices of the location of Wumpus and pit(s)). You should take a screen-shot of at least one such run and include in your report (together with the Agent.h and Agent.cc files, and the readme.txt file). You should also include test scores of your agent for three runs (on different initial configurations), and the average score over those three runs. The cost of each step is per lecture slides and textbook. Recall, each turn counts the same as moving by a single square cell, i.e., has a cost of 1 associated with it. *LeftTurn* means turning 90 degrees counter-clockwise, whereas *RightTurn* means turning 90 degrees clockwise.

2. Consider the following logic puzzle.
Bubba, Raj, Dmitri and Jin like different computer games. One likes Tetris, one likes Pinball, one likes Armageddon, and one likes Checkers. Raj does not like Armageddon or Checkers. Dmitri does not like Armageddon or Tetris. Both Bubba and Dmitri do not like Checkers. Which video game does each person like?

In this problem, you should use Propositional Logic to solve this puzzle. You will use atomic sentences in the form "Likes<P><G>", where <P> is the first letter of the person's name, and <G> is the first letter of the game. Examples: "LikesJC" means Jin likes Checkers, and "¬LikesRA" means Raj does not like Armageddon. Specifically:

a. Write down the negative literals representing the information given in the 3rd, 4th and 5th sentences in the puzzle. Number each literal.
b. Write down a propositional sentence that represents the information that if Bubba, Raj and Dmitri do not like Checkers, then Jin likes Checkers. Convert this sentence to a clause. Number the clause.
c. Use proof by refutation and resolution to show that Jin likes Checkers, i.e., LikesJC. Show each resolution step by indicating the numbers of the two resolvents, and give the resulting clause a number.
d. Use Propositional Logic (PL) inference rules, to show which game Bubba likes (hint: use PL refutation and resolution). You may use the previous clauses and may add sentences to help with the proof, as long as they are consistent with the KB information (= the puzzle description). Convert these sentences to clauses and give each a number. Show each resolution step by indicating the numbers of the two resolvents, and give the resulting clause a (fresh) number.
e. Repeat part (d) for Raj.

3. In this problem, you should use FOL (instead of PL) to formulate the logic puzzle from **Problem2** and then solve it. The atomic sentences will be in the form "Likes(<P>,<G>)", where <P> is either a variable (always lowercase) or one of {Bubba, Raj, Dmitri, Jin}, and <G> is either a variable or one of {Tetris, Pinball, Armageddon, Checkers}. Specifically:

a. Write down the negative literals representing the information given in the 3rd, 4th and 5th sentences in the puzzle. Number each literal.
b. Write down a first-order sentence that represents the information that if Bubba, Raj and Dmitri do not like a game, then Jin likes it. Remember all variables must be quantified. Convert this sentence to a clause. Number the clause.
c. Use proof by refutation and resolution to show that Jin likes Checkers, i.e., Likes(Jin,Checkers). Show each resolution step by indicating the numbers of the two resolvents and any variable substitutions. Give each resulting clause a number.
d. Write down a first-order sentence that represents the information that if a person likes a game, then they do not like any other game. Remember all variables must be quantified. Convert this implication to a clause. Number the clause.
e. Use proof by refutation and resolution to show which game Bubba likes. You should only need to use previous clauses plus a few negative literals indicating that two different games are not equal to each other (e.g., ¬Tetris=Armageddon, ¬Tetris=Pinball, etc.). Show each resolution step by indicating the numbers of the two resolvents, and give the resulting clause a number.