CSCI 5511 - Fall 2021 Final

Due Friday, Dec 17, 11:59 pm

- This is a *take-home* exam. It is open book, open web, but closed people. You may not consult with anyone other than Andy or Ioanna regarding the questions and information on this exam.
- Show all work, clearly and in order, if you want to get full credit. I reserve the right to take off points if I cannot see how you arrived at your answer (even if your final answer is correct).
- If you think a question is unclear or ambiguous, or if you think that there is an error on the exam, either state your assumptions or contact Andy.
- One of the questions has a code component that should be submitted via Canvas. Please comment your code explaining how to run it.
- Your other answers should be submitted as a pdf via Canvas.
- This test has 5 problems and is worth 56 points.
- Good luck!

1	2	3	4	5	Σ
6	10	8	10	24	58

1. Propositional Logic Entailment (*6 points*) For each of the following assertions in propositional logic, either prove that it is true, or else provide a counterexample:

a. (*2 pts*) If $\alpha \models (\beta \vee \gamma)$ then $\alpha \models \beta$ or $\alpha \models \gamma$.

b. (*2 pts*) If $\alpha \models \gamma$ or $\beta \models \gamma$ (or both), then $(\alpha \wedge \beta) \models \gamma$.

c. (*2 pts*) If $\alpha \models (\beta \wedge \gamma)$ then $\alpha \models \beta$ and $\alpha \models \gamma$.

2. Propositional Logic Satisfiability (*10 points*) Consider a sentence in propositional logic in Conjunctive Normal Form (CNF) with exactly k symbols in each group of conjunctions. Such a sentence is defined to be in k-CNF form. For example, a possible sentence in 3-CNF form for a world with 5 symbols (A, B, C, D, E) and 4 clauses is:
$(A \vee \neg B \vee D) \wedge (\neg B \vee \neg D \vee E) \wedge (A \vee D \vee \neg E) \wedge (A \vee C \vee \neg E)$
Given a random k-CNF sentence, it may or may not be satisfiable.
For this problem, write a program named `randompl.py`.

a. (*4 pts*) Write a function `rand3cnf` that takes m and n as parameters, then generates a random sentence in 3-CNF form with m clauses and n symbols. For example, calling this function with $m = 4$ and $n = 5$ might generate the sentence that was given as an example above.
For each clause, you should make certain that literals are not repeated (i.e. you should be able to generate the clauses above, but not a clause of the form $A \vee A \vee B$ or $A \vee B \vee \neg A$)

b. (*4 pts*) Write a function that will call the function from part (a) for m from 30 to 70 (in steps of 10) and n in $[10, 15, 20]$. For each (m, n) setting, generate 100 random knowledge bases and use `SATInterface` to test whether they are satisfiable or not. Have this program print out the results for each.

c. (*2 pts*) For what relationships between m and n is the probability of satisfiability "interesting" (i.e. not 0 or 1)? Describe why you got your answer based on your results from part (b)

3. First-Order Logic (*8 points*) Suppose that you are given the following axioms for a mathematical domain:

- $0 \leq 2$
- $4 \leq 8$
- $\forall x \; x \leq x$
- $\forall x \; x \leq x + 0$
- $\forall x \; x + y \leq y + x$
- $\forall w, x, y, z \; w \leq y \wedge x \leq z \Rightarrow w + x \leq y + z$
- $\forall x, y, z \; x \leq y \wedge y \leq z \Rightarrow x \leq z$

a. (*4 pts*) Show a forward-chaining proof of the sentence $4 \leq 2 + 8$. (Use only the axioms here, not any other facts you may think you know about arithmetic) Only include steps that lead to success, not irrelevant steps.

b. (*4 pts*) Show a backward-chaining proof of the sentence $4 \leq 2 + 8$. Only include steps that lead to success, not irrelevant steps.

4. Planning (*10 points*) Consider the following problem:
An agent is trapped in a puzzle room! To escape from the room, the agent must get the exit door to open.
The escape room uses cleverly rigged bowls that can sense when metal marbles are placed in them, which can cause doors to open or close.

- There are three bowls, Bowl1, Bowl2 and Bowl3 in the room.
- There are two doors, ExitDoor and ClosetDoor that lead out of the room.
- There is a single marble (Marble1) in Bowl1.
- Putting a marble in Bowl3 will open the Closet Door to reveal another bowl (Bowl4) that also contains a single marble (Marble2.) (Removing all marbles from Bowl3 will cause the closet door to close again.)
- Putting one marble in Bowl1 and one marble in Bowl2 will open the exit door. (Removing marbles so that either Bowl1 or Bowl2 is empty will cause the exit door to close again.)
- You may put as many marbles in a bowl as you wish. You can safely assume that marbles only need to be in bowls.

Using the following predicates

- In(x, y)
- Door(x)
- Open(x)
- Bowl(x)

and any other predicates that you deem necessary, answer the following questions:
(You may also want to add and define some constants that represent objects)

a. (*2 pts*) What is the initial state for this planning problem?

b. (*2 pts*) What is the goal state for this planning problem?

c. (*6 pts*) What are the PDDL action schema that are required for an agent to use planning to find a solution?

5. Problem Approaches (*24 points*) For each of the following problems, consider the two given approaches, and give enough implementation detail for each so that you can argue why one approach is better than the other.
I expect a brief (2-3 sentence) description of how each approach will work, followed by a short (4-5 sentence) argument about which one is better. Arguments should include runtime and/or memory estimates when comparing the approaches.

a. (*8 pts*)
Knight's tour - Find a way for a knight to visit all 64 squares on a chessboard without visiting the same space twice.

- Uninformed Search
- Planning

b. (*8 pts*)
Crossword puzzle - Figure out how to put english words into a grid-like structure that forms words going across and down. (For those who have encountered crossword puzzles before, you can ignore the clues part of this and just consider putting a set of legal words into the grid as your goal.)

- Local Search
- Constraint Satisfaction

c. (*8 pts*)
Minesweeper agent solver - Write an agent that can solve a minesweeper game. (As described in the *Gameplay* section of `https://en.wikipedia.org/wiki/Minesweeper_(video_game)`)

- Knowledge-Based Agent
- Goal-Based Agent