CS 188 Summer 2020 Introduction to Artificial Intelligence — Assessment 1

Due: June 29, 2020 at 12:29 PM (submit via Gradescope before lecture starts).

Policy: Must be solved individually. No collaboration allowed.

Submission: Your work must be legible, and work for each question must start on a new page. Do not put work for multiple problems on the same page. When submitting to Gradescope, please assign pages correctly to each problem.

First name	
Last name	
SID	

For staff use only:

Q1.	Consistent Heuristics	/10
Q2.	Now Inconsistent	/10
Q3.	Search	/50
Q4.	More Search	/20
	Total	/90

Q1. [10 pts] Consistent Heuristics

Given a state space S consisting of states $s_0, s_1, ..., s_g$ where s_g is the goal state, a heuristic h, the true optimal cost-to-go function h^*, and a cost function for edges c, prove or provide a counterexample: Consistent heuristics are admissible. NOTE: you WILL LOSE POINTS if you do not follow the given notation.

Q2. [10 pts] Now Inconsistent

Pacman hasn't had much to do recently, so he's been playing games of Solitaire on his own to pass the time. Unfortunately, he's not very good at it. In order to help himself get better, Pacman tries to model this question as a search problem with the states being any configuration of cards. The cost of edges between states, modeled by function c, is the number of moves closer Pacman gets to winning. An optimal cost-to-go function, h^*, will tell Pacman how many moves away from winning he is given any state. Given h^*, design an admissible but inconsistent heuristic h and prove that it is indeed admissible but inconsistent. NOTE: you do not need to understand Solitaire to solve this problem. (However, and only if it helps you formulate your solution, you may answer this question for any Solitaire variant or rule set of your choosing.)

Q3. [50 pts] Search

To discuss a strategy to play against Pacman, the ghosts send each other messages in English, encryped by a bijective mapping from the alphabet (a-z) to a permutation of the alphabet. Pacman intercepted an encrypted message of the ghosts, which is a string consisting of lower case letters (a-z) and spaces. Let this string be X, and assume no missing or extraneous spaces.
Pacman decided to cast this problem as a search problem: he is searching for a mapping (equivalently, a length-26 sequence of letters) that maps the encrypted message X to a readable paragraph. **Each search state is a mapping.** Pacman assembled set W, a sufficiently large collection of English words that covers the ghosts' vocabulary. But the ghosts make some typos in their messages.

(a) Goal test

(i) [5 pts] Provide a reasonable goal test in terms of X and W that generally works despite that there are typos. Then, explain the rationale of your goal test in at most 2 sentences.

(ii) [5 pts] Assume that the original message (before the ghosts encrypted it to X) does not have typos and contains all the letters in the alphabet.
Is it guaranteed that you will find the correct mapping (the one that the ghosts are actually using) with your goal test?
If so, explain your reasoning in 2 sentences; otherwise, give an example of a letter in the original message that is hard to get correct and explain your reasoning in 1-2 sentences.

(iii) [5 pts] Describe a scenario that your goal test does not work, and explain why. You should complete your answer in no more than 3 sentences.

(b) Recall that each search state is a mapping. For part **(b)**, assume perfect goal test and that we are using **tree search**.

(i) [10 pts] Give a start state and a set of legal actions such that DFS is guaranteed to reach a goal state. Explain in at most 4-5 sentences, why DFS is guaranteed to reach a goal state for your proposed start state and legal actions.

(ii) [5 pts] What is the time complexity of running DFS on your proposed formulation of search?

(iii) [5 pts] For your proposed start state and legal actions, is BFS guaranteed to reach a goal state? Explain in 1-2 sentences.

(c) [15 pts] Recall that Pacman is searching for a mapping that decodes the encrypted message X to a readable paragraph.
Pacman found a 26 by 26 matrix $[a_{ij}]$, where a_{ij} is the probability that the ith character is followed by the jth character in English words. He decided to formulate the search as follows:
The start state is the mapping that maps a-z to itself (not decoding anything). A legal action is to swap two characters in the decoding mapping (the decoder). For example, from the start state, swapping a and b results in a decoder that maps the alphabet to "bacde...", which is a legal action. This decoder swaps only a's and b's and leaves everything else unchanged.

Help Pacman complete this search problem by completing the following:

- Provide a non-trivial cost function and a non-trivial heuristic
- Discuss how the heuristic could meaningfully guide A* search
- Explain whether the heuristic is admissible and/or consistent

Q4. [20 pts] More Search

Consider the following problem setup:
The head TA is figuring out the discussion schedule. We have M TAs and N discussion sections. Each discussion section has a known time slot, and fortunately, at most 2 discussions are in the same time slot. In the N discussion sections, we need exactly K of them to be exam prep sections, and the rest to be normal sections.
We also want the following conditions to hold:

- We would like to have exactly one TA for each discussion section.
- A TA cannot be teaching 2 discussion at the same time slot.
- Some TAs do not teach regular discussions, some do not teach exam prep, and others can teach either.
- If two discussions are in the same time slot, we also don't want them to be the same type of discussion for the sake of best allocation of resources.

(a) [10 pts] Cast this as a search problem. Specifically, appropriately define the state representation, the successor function, the start state and the goal test.

State representation:

Successor function:

Start state:

Goal test:

(b) [10 pts] Calculate the state space size and the branching factor of the search tree, then suggest a search algorithm to tackle the problem. If you are using DFS/BFS, explain your choice of algorithm in a paragraph. If you are using UCS/Greedy/A* search, please define in a paragraph what cost function and/or heuristic is being used. It doesn't necessarily need to be a rigorous definition but more of a qualitative description of what the cost function/heuristic does.

State space size: _____

Branching factor: _____

Proposed search algorithm: _____

Explanation: