Portland State University

CS 441/541: Artificial Intelligence, Winter 2022

Homework #1

Note: This assignment is **due Sunday, 1/23 @ 10pm.** Please email your typed or written (scanned) homework solutions to our TA by the due date. Write and exposit clearly – show a sufficient amount of work on each problem, without over indulging, please.

Each student will turn in an individual assignment (so that we have something upon which to base your individual grade). However, you are encouraged to discuss and work through these problems with your instructor, TA and above all, other students in our class. You will not consult the internet for hints/answers – (you are nevertheless encouraged to consult our lecture notes and wiki/academic texts to help further elucidate any concepts that you find challenging). **Bottom line**: the answers and work you submit will authentically be your own.

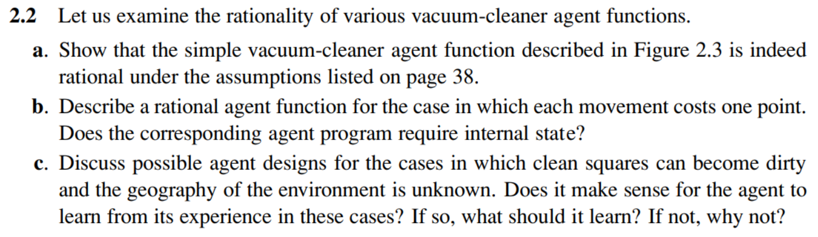
1. Given any invertible, real-valued, *mxn* matrix *A*, prove that: $\left(A^T\right)^{-1} = \left(A^{-1}\right)^T$. In other words, show that the inverse of the transpose of a matrix is equal to the transpose of its inverse. (Hint: you may consult our course notes; it is helpful to know that $\left(AB\right)^T = B^T A^T$, which you need not prove).

2. After your yearly checkup, the doctor has bad news and good news. The bad news is that you tested positive for a serious disease, and that the test is 99% accurate (i.e. the probability of testing positive given that you have the disease is 0.99, as is the probability of testing negative give that you don't have the disease). The good news is that the disease is very rare, afflicting only one in 10,000 people on average. What are the chances that you actually have the disease? (Show your calculations as well as giving the final result).

3. Using techniques from Calculus, show directly that the maximum value of a 1-D Gaussian distribution occurs at the point x = μ.

4. (i) What is the expected number of heads in four flips of a fair coin? (show your work). (ii) Suppose a fair coin is tossed repeatedly until a head is observed for the first time. What is the expected number of tosses required? (show your work – note that the number of flips until we observe a head could be arbitrarily large).

5. Do problem 2.2 from Chapter 2 in our text (p. 61).



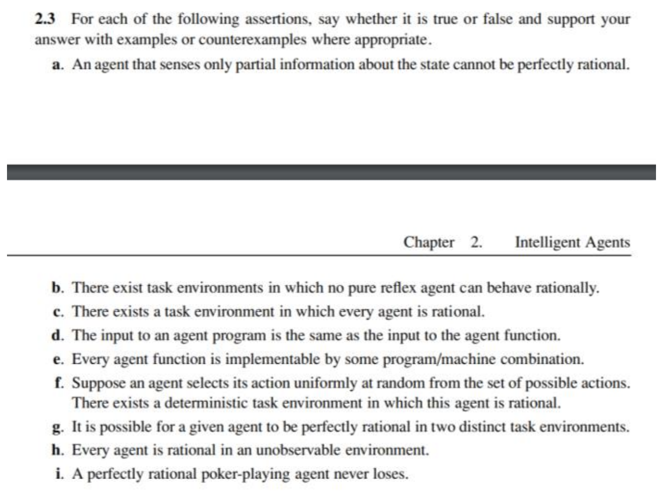
2.2 Let us examine the rationality of various vacuum-cleaner agent functions.

a. Show that the simple vacuum-cleaner agent function described in Figure 2.3 is indeed rational under the assumptions listed on page 38.

b. Describe a rational agent function for the case in which each movement costs one point. Does the corresponding agent program require internal state?

c. Discuss possible agent designs for the cases in which clean squares can become dirty and the geography of the environment is unknown. Does it make sense for the agent to learn from its experience in these cases? If so, what should it learn? If not, why not?

6. Do problem 2.3.


2.3 For each of the following assertions, say whether it is true or false and support your answer with examples or counterexamples where appropriate.

a. An agent that senses only partial information about the state cannot be perfectly rational.

b. There exist task environments in which no pure reflex agent can behave rationally.

c. There exists a task environment in which every agent is rational.

d. The input to an agent program is the same as the input to the agent function.

e. Every agent function is implementable by some program/machine combination.

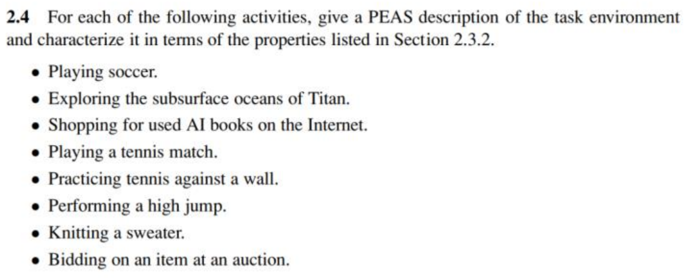
f. Suppose an agent selects its action uniformly at random from the set of possible actions. There exists a deterministic task environment in which this agent is rational.

g. It is possible for a given agent to be perfectly rational in two distinct task environments.

h. Every agent is rational in an unobservable environment.

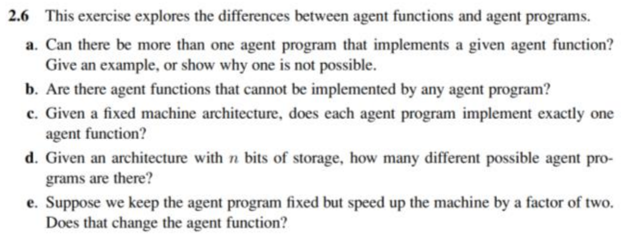
i. A perfectly rational poker-playing agent never loses.

7. Do problem 2.4.


2.4 For each of the following activities, give a PEAS description of the task environment and characterize it in terms of the properties listed in Section 2.3.2.

- Playing soccer.
- Exploring the subsurface oceans of Titan.
- Shopping for used AI books on the Internet.
- Playing a tennis match.
- Practicing tennis against a wall.
- Performing a high jump.
- Knitting a sweater.
- Bidding on an item at an auction.

8. Do problem 2.6.


2.6 This exercise explores the differences between agent functions and agent programs.

a. Can there be more than one agent program that implements a given agent function? Give an example, or show why one is not possible.

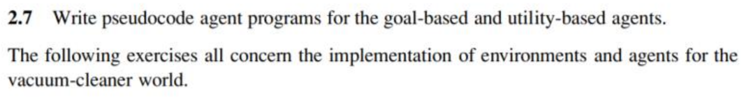
b. Are there agent functions that cannot be implemented by any agent program?

c. Given a fixed machine architecture, does each agent program implement exactly one agent function?

d. Given an architecture with n bits of storage, how many different possible agent programs are there?

e. Suppose we keep the agent program fixed but speed up the machine by a factor of two. Does that change the agent function?

9. Do problem 2.7.


2.7 Write pseudocode agent programs for the goal-based and utility-based agents.

The following exercises all concern the implementation of environments and agents for the vacuum-cleaner world.

10. For this problem you will implement several agent types in a variant of the "vacuum world" described in our text/lecture. You are welcome to write code in any language you prefer (you need not turn the code in with your homework) – your implementation can have graphics, but again, this is not required.

Consider a 3x3 grid for the vacuum world environment. The agent can go up, down, left right, suck or do nothing. Randomize the starting location of the robot, the number of rooms with dirt piles (let's say in separate cases: 1, 3 and 5 piles initially, respectively), as well as the locations of the dirt piles. Record and compare the performance score (e.g. total number of moves to clean the environment – you are free to make up your own performance score/evaluation standard). You should cap the number of steps in your simulations reasonably to avoid infinite loops. Report results for the following types of agents:

(i) A simple reflex agent (in this case define a "rule table" based on the location/dirt present – just as in the book); in this case you can just make up a rule for where the robot moves next, according to its current location. Please note that the "where to move next" rule is only based on the current percept and should be fixed by the rule table (which is to say we don't actively "learn" a movement strategy).

(ii) A randomized agent (movement and suck/no suck are randomized). Describe explicitly how you randomize the agent.

(iii) Apply murphy's law with the same environment for a reflex agent. Murphy's law: 25% of the time the suck action fails to clean the floor if it is dirty and deposits dirt onto the floor if the floor is clean; suppose also that the "dirt sensor" give the wrong answer 10% of the time.

(iv) Apply murphy's law conditions (as in iii) for a randomized agent.

For your answer, write a short summary of your findings in no more than a few paragraphs. Please report/contrast your results for each agent and environment for parts (i)-(iv); in each case, report the average performance, in addition to the performance measure you used. So that the results are meaningful for comparison, run each agent for a fixed number of initial dirt piles (1, 3 and 5, respectively), with random initial locations for the piles and agent; report results for a reasonable number of trials in each case (e.g. 100 or so). Offer some insightful commentary.

In summary: in the end, you should have comparative performance measures for each of the (4) agents, for cases of 1, 3 and 5 initial dirt piles (12 cases in all), respectively; run each experiment 100 or so times.

For clarity: Example result table might look something like this:

Agent Type/Initial dirt piles	1	3	5
(i)	Performance measure…	…	…
(ii)	…	…	…
(iii)	…	…	

(iv)	…	…	…