CS 440 Practice Final Problems
Dec 17, 2015

Rules

- Write your answers directly on these pages.
- You may use a single sheet of paper with reference materials (two sides) that you have prepared.
- You may not discuss the test with anyone else or use any electronic devices (the problems involve minimal calculation).
- You may ask clarification questions during the exam. Raise your hand and we will come to you.
- The test has 7 problems. All count equally.
- Show your work for partial credit.

Your name: ________________________

When you get the exam, you will see something like this. The problems that follow include more than you would see on an actual exam (at least twice as many), and have not been formatted to allow you to write the answers in place. These are real problems. They illustrate the style of the exam—questions that ask you to apply concepts and definitions from class to new problems and cases, and questions that ask you to work out the predictions of algorithms and models from class to specific new examples. This inventory of problems reflects the slightly different emphasis of the course in previous years, so you can expect fewer questions on stuff we didn't spend so much time on and maybe some new questions on stuff that we covered carefully that's not represented here.

1. (Agents) One way to describe the reinforcement learning algorithms we learned is that they compile experience of the world into a program for a reactive agent that acts effectively in that world. Give an example of a world in which a reactive agent cannot act effectively and describe what goes wrong (mathematically and practically) if you try to use reinforcement learning to build an agent for your example.

2. (Agents) In discussing face detection, we saw boosting, a machine learning technique that combines simple, inaccurate classifiers into a single, complex, accurate classifier. A classifier created using boosting has similar advantages to the subsumption architecture: both involve simple rules that operate independently and extensibly, with results that are combined together into an effective system. However, the subsumption architecture is designed to choose actions, not make classification decisions. Why don't reinforcement learning algorithms (value or policy iteration or temporal difference or Q-learning) naturally lend themselves to learning a subsumption architecture?

3. (Search) You are given a chemical formula: a mapping from chemical elements to counts. (The domain of the mapping is the set of elements that occur in the molecule. The range is the number of atoms of that element that occur in the molecule. For example, H_2O.) Your job is to find possible molecular structures. To a first approximation, molecular structures can be represented as graphs. Each vertex represents an atom and is labeled with a chemical element and each edge represents a bond and is labeled with an integer (single bond, double bond, etc). In a stable molecule, the sum of the labels on the edges for each vertex must equal the chemical valence of the vertex element. Describe a formal search problem to compute possible molecular structures for a chemical formula. Briefly, what are the states, what are the actions, what are the transitions, what is the initial state, what is a goal? (Your answer can be short English sentences; mathematical definitions are not required. Chemistry could assign a cost to a graph in terms of its energy—but this is beyond the scope of this class.)

4. (Markov Decision Processes) You are building a robot to navigate to targets in a 3x3 grid world. The robot knows its position in the grid. It starts at S, gets a reward of 10 units when it reaches a specified square but loses 10 if it winds up in another specified square. After the reward the exercise stops. It costs 1 unit to move each step. The discount is 1 (so future rewards are just as good as present rewards).
(a). Suppose the agent moves with perfect accuracy one square in the direction of its choice. Indicate an optimal policy for the agent below.

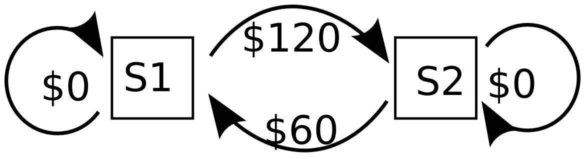
	-10	
S		+10

(b). Suppose that the agent can try to move one square in the direction of its choice, but the effects of action are unpredictable and the agent has a sizeable chance of slipping. When the agent slips it's equally likely to move to any adjacent square. Indicate an optimal policy for the agent below.

	-10	
S		+10

(c). Can these policies be implemented in reactive agents? Explain your answer.
(d). It comes time to actually commit to a design for your robot. Should you implement the first policy, the second policy, or some other strategy? Justify your answer.

5. (Markov Decision Processes) Consider a world with two states. When you go from S_1 to S_2 you get \$120. When you go from S_2 to S_1 you get \$60. When you stay you get \$0. Your discount factor is $\frac{2}{3}$. The situation is as shown below:



Suppose you choose an action in S_1 that leads you to S_2 with probability $\frac{1}{3}$ and also choose an action in S_2 that leads you to S_1 with probability $\frac{1}{3}$. Compute the value function (expected utility over the indefinite future) for S_1 and S_2.

6. (Games) A classic illustration of the concepts of game theory is a game known as the prisoner's dilemma. There are two players (A and B). They have two actions: cooperate (C) or defect (D). The payoffs are given below:

		A	
		C	D
B	C	$A: 2, B: 2$	$A: -1, B: 5$
	D	$A: 5, B: -1$	$A: 0, B: 0$

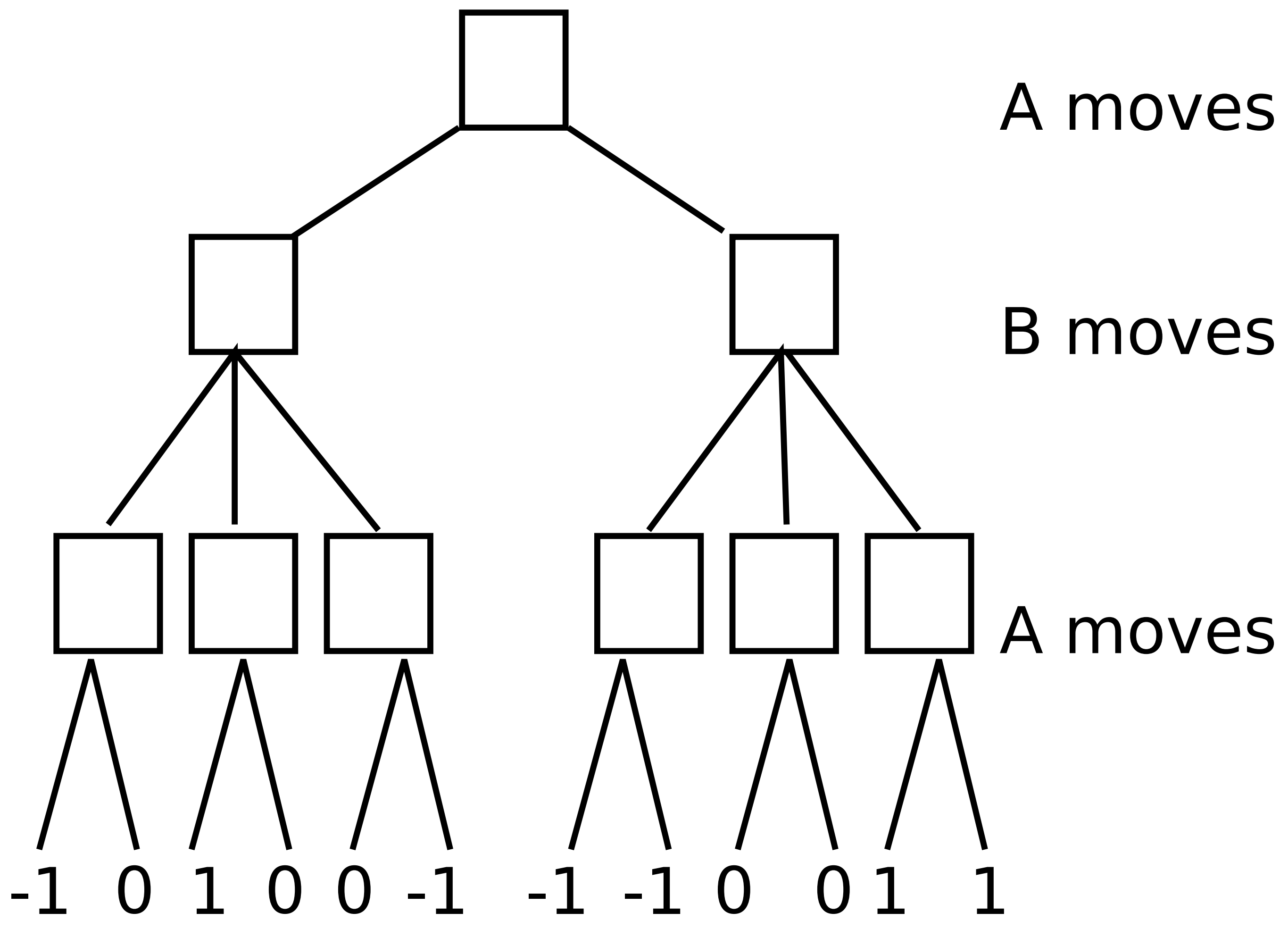
(a). There is one equilibrium. What is it?
(b). Two players can outperform the equilibrium, but only if they make flawed decisions. How?
(c). Suppose you wanted to use learning and do well both against perfect reasoners and flawed reasoners: what strategy would you choose? For example, do you think you could start out playing optimally and then learn to do better against a flawed player? Or should you start out with a flawed strategy and play optimally if your partner is exploiting you? Hint: what if your opponent is learning the same way you are?

7. (Games) Here is a zero-sum two-player game. The players are A and B. The players each choose actions C and D simultaneously and get the following payoffs:

		A	
		C	D
B	C	$A: 2, B: -2$	$A: -2, B: 2$
	D	$A: -1, B: 1$	$A: 3, B: -3$

What is the equilibrium for this game?

8. (Games) The tree below describes a zero-sum turn-taking game with two players, A and B. A moves first, then they alternate. The values at the leaves shows the score for A.



(a). *Write the values for A in each internal node*, assuming both A and B play optimally.
(b). Suppose A is searching for the best strategy, and the search considers nodes from left to right in the order they are diagrammed above. *Write an X next to internal and leaf nodes that don't need to be searched* because they are known not to affect the value of their subtree at the time they are explored.

9. (Learning) A driving robot uses a nearest neighbor model and some training data to predict the maximum speed of a vehicle from a measurement of its size (say, the area in square meters occluded by the car when viewed from behind). You assess the performance of the model by the percentage of the time you get within 10% of the true value. In other words, you want to be very close as often as possible, but it doesn't matter how wrong you are if you're wrong.
You build an initial version of the system and you find many examples in test data where performance is worse considering two neighbors than considering just the closest one. A coworker suggests that if you get enough data, you will eliminate these cases and predicting from two neighbors will work better on all points. Is this necessarily true? Explain.

10. (Learning) Consider an environment with many agents, acting independenty. You want to learn how to act effectively. You can perceive some set of feature values F in your surroundings. These feature values are enough to determine a good reactive policy. You can observe the actions of other agents, and you can also tell what features those agents should perceive from their perspective. Assume that other agents have similar preferences to yours and act effectively in pursuit of them.
(a). Describe a way of using supervised learning to decide what to do, by imitating other agents. What is the exact problem you want to learn, how do you get your training data, how do you represent your training data, and what is your test data?
(b). For (a) we made a bunch of assumptions: others are like you and act well, others have a reactive policy, you can take their perspective. How might supervised learning by imitation fail if these assumptions are violated?
(c). How would you design an agent to act well in this scenario using reinforcement learning (for example, Q learning)?
(d). Reinforcement learning does not require the assumptions of imitation learning. But what considerations would argue in favor of imitation learning over reinforcement learning when both apply?