Homework #3
CS 4710: Artificial Intelligence (Spring'20)

Due Tuesday 03/03 11:59 pm

Introduction: Welcome to the third official homework for CS4710! As you are hopefully already aware, this PDF comprises the written component of the third problem set. In addition to solving the problems found below, you will also need to complete the coding part of the assignment. Finally, we'd like to remind you that while you are allowed a partner for the coding part of the assignment, you are **NOT** allowed a partner for this and all future written components. All written work should be yours and yours alone. This being said, in addition to being able to ask questions at office hours, you are allowed to discuss questions with fellow classmates, provided 1) you note the people with whom you collaborated, and 2) you **DO NOT** copy any answers. Please write up the solutions to all problems independently.

Collaborators:

Problem 1 (Markov Decision Processes) – 6 Points: Annie is a 5-year old girl who loves eating candy and is ambivalent regarding vegetables. She can either choose to eat candy (Hershey's, Skittles, Peanut Butter Cups) or eat vegetables during every meal. Eating candy gives her $+10$ in happiness points, while eating vegetables only gives her $+4$ happiness points. But if she eats too much candy while sick, her teeth will all fall out (she won't be able to eat any more). Annie will be in one of three states: healthy, sick, and toothless. Eating candy tends to make Annie sick, while eating vegetables tends to keep Annie healthy. If she eats too much candy, she'll be toothless and won't eat anything else. The transitions are shown in the table below.

Health condition	Candy or Vegetables?	Next condition	Probability
healthy	vegetables	healthy	1
healthy	candy	healthy	1/4
healthy	candy	sick	3/4
sick	vegetables	healthy	1/4
sick	vegetables	sick	3/4
sick	candy	sick	7/8
sick	candy	toothless	1/8

(a) (1 Point) Model this problem as a Markov Decision Process: formally specify each state, action, and transition $T(s, a, s')$ and reward $R(a)$ functions.

(b) (1 Point) Write down the Value function $V(s)$ for this problem in all possible states under the following policies: π_1 in which Annie always eats candy and π_2 in which Annie always eats vegetables. The discount factor can be expressed as γ.

(c) (1 Point) Start with a policy in which Annie always eats candy no matter what the her health condition is. Simulate the first two iterations of the policy iteration algorithm. Show how the policy evolves as you run the algorithm. What is the policy after the third iteration? Set $\gamma = 0.9$.

(d) (3 Points) Which of the following five statements are true for an MDP? Select all that apply and briefly explain why.

   (i) If one is using value iteration and the values have converged, the policy must have converged as well.

   (ii) Expectimax will generally run in the same amount of time as value iteration on a given MDP.

   (iii) For an infinite horizon MDP with a finite number of states and actions and with a discount factor that satisfies $0 < \gamma <= 1$, policy iteration is guaranteed to converge.

   (iv) There may be more than one optimal value function.

   (v) There may be more than one optimal policy.

Solution 1:

Problem 2 (Reinforcement Learning) – 6 Points:

(a) (1 Point) In class we learned a couple of temporal-difference techniques for reinforcement learning. Now we'd like to take the next step (literally). Suppose we take two steps and get the state/action sequence s-a-s'-a'-s". Write temporal-difference update equations in terms of V.

(b) Consider the following deterministic Transition/Reward Model for an MDP with states (S1, S2, S3, S4, S5, S6) and actions (A1, A2, A3):

From	Action	To	Reward
S1	A3	S2	3
S2	A1	S1	2
S2	A2	S3	1
S3	A1	S4	2
S3	A2	S5	10
S4	A3	S3	5
S5	A1	S6	7
S6	A3	S5	2

   (i) (1 Point) Suppose we start in state S3. We run Q-learning on this MDP using a greedy policy (always choose action with best Q-value). Ties are given to the action with the lower number (A1 > A2, etc). Assume $\alpha = 0.5$ and $\gamma = 0.9$. What are the first 4 (state, action) pairs visited, including the start state and the following action?

   (ii) (1 Point) Why is this simple-greedy policy limited and what can we alter about this algorithm to overcome this limitation?

(c) Consider the following Transition/Reward Model for an MDP with states (S1, S2, S3, S4, S5) and actions (A1, A2) – (S4, S5) are terminal states with no valid actions. Assume $\gamma = 1.0$:

From	Action	To	Reward	Probability
S1	A1	S2	0	0.5
S1	A1	S3	0	0.5
S1	A2	S2	0	0.25
S1	A2	S3	0	0.75
S2	A1	S4	6	1.0
S2	A2	S4	12	0.5
S2	A2	S5	4	0.5
S3	A1	S5	16	1.0

   (i) (1 Point) Compute $Q^*(s, a)$ for each state-action pair using Q-value iteration and show the process.

   (ii) (1 Point) Compute the optimal policy.

(iii) (1 Point) Initialize Q to 0 for all (state, action) pairs for q-learning. With $\alpha = 0.5$, compute Q after seeing: [(S1,A1,S2) -> (S2,A2,S4)] [(S1,A2,S3) -> (S3,A1,S5)] [(S1,A2,S2) -> (S2,A1,S4)]

Solution 2:

Problem 3 (Beam Search) – 2 Points: Ant colony optimization is an optimization technique that was inspired by the foraging behavior of real ant colonies. When searching for food, ants initially explore the area surrounding their nest in a random manner. Ants communicate indirectly by means of chemical pheromone trails, which help them find the shortest paths between their nest and the nearest food. While moving, they leave a trail of chemical pheromone behind them. Once an ant finds food, they vary the amount of pheromone they leave depending on the quality and quantity of food. This indirect communication between ants via pheromone trails enables them to find the shortest paths between their nest and the food sources.
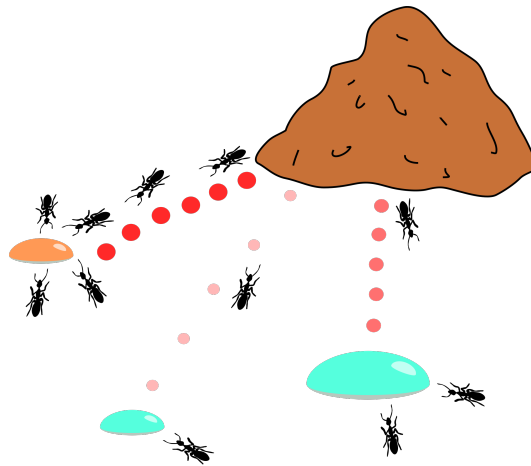
Describe (in words) how the ant colony optimization problem can be modeled through a local beam search algorithm. Indicate what k stands for and how the ants find the k-best successors for each iteration.

Solution 3: