CS 4710: Artificial Intelligence **Spring 2020**

Homework 3

Student Name: David Chen *Computing ID: dzc5ta*

- You will need to submit your solutions in PDF to UVA-Collab.

I collaborated with Timothy Han (txh7es)

1. a)
   States: healthy, sick, and toothless
   Actions: Eat candy or eat Vegetables

   T(healthy, vegetables, healthy) = 1
   T(healthy, candy, healthy) = 1/4
   T(healthy, candy, sick) = 3/4
   T(sick, vegetables, healthy) = 1/4
   T(sick, vegetables, sick) = 3/4
   T(sick, candy, sick) = 7/8
   T(sick, candy, toothless) = 1/8

   R(candy) = +10 happiness
   R(vegetables) = +4 happiness

   b)
   π_1: Annie always eats candy.
   V(s) =$\sum$ T(s, a, s')[R(s, a, s') + γV^{π_1}(s')]
   V^{π_1}(healthy) = T(healthy, candy, healthy)[R(healthy, candy, healthy) + γV^{π_1}(healthy)] + T(healthy, candy, sick)[R(healthy, candy, sick) + γV^{π_1}(sick)]
   V^{π_1}(healthy) = 1/4 * (10 + γV^{π_1}(healthy)) + 3/4 * (10 + γV^{π_1}(sick))
   V^{π_1}(healthy)) = 53.8899

   V(sick) = T(sick, candy, sick)[R(sick, candy, sick) + γV^{π_1}(sick)] + T(sick, candy, toothless)[R(sick, candy, toothless) + γV^{π_1}(toothless)]
   V(sick) = 7/8 * (10 + γV^{π_1}(sick)) + 1/8 * (10 + γV^{π_1}(toothless))
   V^{π_1}(sick)) = 47.0588

   V(toothless) = 0

   π_2: Annie always eats vegetables.
   V(s) =$\sum$ T(s, a, s')[R(s, a, s') + γV^{π_2}(s')]
   V^{π_2}(healthy) = T(healthy, vegetables, healthy)[R(healthy, vegetables, healthy) + γV^{π_2}(healthy)]

V^{π_2}(healthy) = 1 * (4 + γV^{π_2}(healthy))
V^{π_2}(healthy)) = 40

V^{π_2}(sick) = T(sick, vegetables, healthy)[R(sick, vegetables, healthy) + γV^{π_2}(sick)] + T(sick, vegetables, sick)[R(sick, vegetables, sick) + γV^{π_2}(toothless)]
V^{π_2}(sick) = 1/4 * (4 + γV^{π_2}(healthy)) + 3/4 * (4 + γV^{π_2}(sick))
V^{π_2}(sick)) = 40

c)
π_1: Annie always eats candy.
V^{π_1}(toothless)] = 0
V^{π_1}(sick)) = 47.0588
V^{π_1}(healthy)) = 53.8899

$\pi^*(sick)$: candy: 7/8 * (10 + .9 * 47.0588) + 1/8 * (10 +.9*0) = 47.058805
vegetable: 3/4 * (4 + .9 * 47.0588) + 1/4 * (4 + .9 * 53.8899) = 47.8899175
$\pi^*(sick)$ = eat vegetable

$\pi^*(healthy)$: candy: 3/4 * (10 + .9 * 47.0588) + 1/4 * (10 + .9 * 53.8899) = 53.8899175
vegetable: 1 * (4 + .9 * 53.8899) = 52.50091
$\pi^*(sick)$ = eat candy

π_2: Annie eats vegetables when she is sick and candy when she is healthy.
V^{π_2}(toothless)] = 0
V^{π_2}(sick)) = 51.1685
V^{π_2}(healthy)) = 57.1685

$\pi^*(sick)$: candy: 7/8 * (10 + .9 * 51.1685) + 1/8 * (10 +.9*0) = 50.29519375
vegetable: 3/4 * (4 + .9 * 51.1685) + 1/4 * (4 + .9 * 57.1685) = 51.40165
$\pi^*(sick)$ = eat vegetable

$\pi^*(healthy)$: candy: 3/4 * (10 + .9 * 51.1685) + 1/4 * (10 + .9 * 57.1685) = 57.40165
vegetable: 1 * (4 + .9 * 57.1685) = 55.45165
$\pi^*(sick)$ = eat candy

π_3: Annie eats vegetables when she is sick and candy when she is healthy.

The optimal policy has been found since it stayed the same. Therefore, the policy after the third iteration will still be: Annie eats vegetables when she is sick and candy when she is healthy.

d)
i) Policy converges long before values converge.
iii) There must be a finite number of policies, since there is a finite number of states and actions.
v) If there are two actions that lead to the same state than either action can be in an optimal policy.

2. a)
V^{π}(s') = V^{π}(s') + α(R(s',π(s'), s") + γV^{π}(s"))
V^{π}(s) = V^{π}(s) + α(R(s,π(s), s') + γ(V^{π}(s') + α(R(s',π(s'), s") + γV^{π}(s")))

b)
i) The first four (state, action) pairs would be:
(S3, A3) $\rightarrow$ $(S5, A1) \rightarrow (S6, A3) \rightarrow (S5, A1)$

ii) If only the best Q-value is chosen, there might be unexplored regions. We can alter this by sometimes, choosing a sub-optimal option. This way, other regions can be explored.

$Q^*(s, a) = \Sigma_{s'} T(s, a, s')[R(s, a, s') + \gamma max_{a'} Q^*(s', a')]$
$Q^*(S3, A1) = 1 * 16 = 16$
$Q^*(S2, A2) = 0.5 * 4 + 0.5 * 12 = 8$
$Q^*(S2, A1) = 1 * 6 = 6$
$Q^*(S1, A2) = 0.25 * 8 + 0.75 * 16 = 14$
$Q^*(S1, A1) = 0.5 * *8 + 0.5 * 16 = 12$

The optimal policy: At state S1 and S2, take action A2 since that action maximizes the value at those states.

$Q = 0$
$Q \leftarrow (1 - 0.5)(0) + 0.5 * 12 = 6$
$Q \leftarrow (1 - 0.5)(6) + 0.5 * 16 = 11$
$Q \leftarrow (1 - 0.5)(11) + 0.5 * 6 = 8.5$

3. In beam search, there there are k beams, or best paths, at the end of each iteration. For each iteration, each beam is expanded and different paths are explored. Each path is then ranked and the best k paths become the new starting beams. This process repeats until a goal it reached. Beam search can model the ant colony optimization problem because each beam can be a path that is being explored. The ants will when leave more pheromone based on how much potential the food on that path has. The amount of pheromone on a path can be used to rank each path. At the end of the day ants return to their nest after a long day of gathering food. At the beginning of the next day, the ants start exploring from the top k paths with the most pheromone.