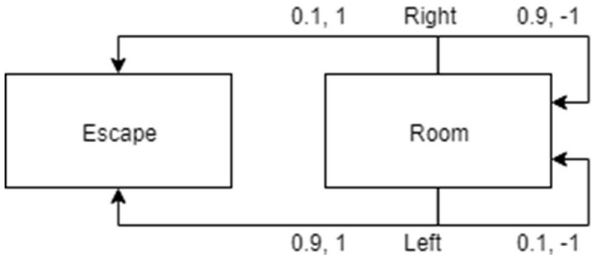
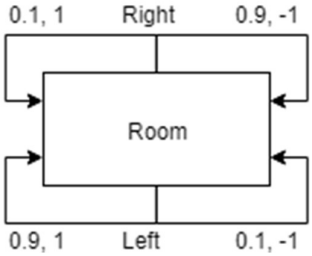
Mock Exam Solutions

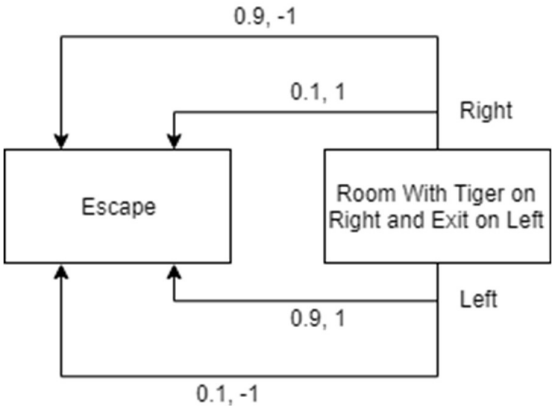
1. a)i) Assume tiger reward is -1 and escape reward is 1. I will also make the assumption that the bunny immediately retreats to the initial room after moving to the tiger room and immediately leaves after reaching the room with the exit in it.



ii) If the exit now leads back to the starting state then all actions will now lead back to the starting state.



If the tiger now ends the MDP then all actions will lead to the absorbing “Escape” state.

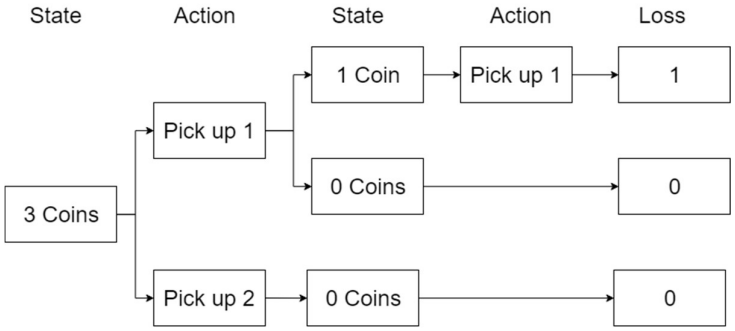


The rewards and probabilities aren’t changed by either.

b)i)Model-based
ii)Reward shaping is us tailoring the reward function based on our knowledge of the task to guide our agents towards optimal solutions faster. The problem is that we are introducing our own idea of what the optimal solution looks like into the task and so it may lead to results that would be suboptimal in the original, unshaped task.

iii) Behavioural Cloning seeks to find a policy that best describes the policy used in the training data. Inverse Reinforcement Learning seeks to find the reward distribution implied by the training data.

2. a)i) The next possible states are “One coin” and “Zero Coins” as the next player will pick up one or two of the coins.
   ii) We can play this game multiple times, performing the same action and recording the state after the adversary has taken their turn. We can then use the results to estimate the transition function.
   iii) Assuming a loss of 1 for a losing the game:



   If our policy is pick up one coin, the maximum possible loss is 1
   If our policy is pick up two coins, the maximum possible loss is 0
   So, the Mimimax policy, that minimises the maximum possible loss, would be to pick up two coins.

   b)i) V = **ws**. V is the scalar approximation of the Value of the state. **w** is a feature weight vector. **s** is the state feature vector.

   c)$\gamma = 1$
   i) First visit R for A is 1+1=2 and -1. So V for A is (2-1)/2 = 0.5
   ii) Every visit R for A is 1+1=2, 1, and -1. So V for A is (2+1-1)/3 = 0.666…
   iii) B two step return is 1+-1 = 0. A two step return is -1+0 = -1

3. a)i) Bootstrapping means that it uses pre-existing estimates to create new estimates.
ii) Dynamic Programming Methods.

b)i) There will now be three states: A room with a tiger to the right and exit to the left s_0, a room with an exit to the right and a tiger to the left s_1, and an absorbing state s_2. Our set of states is S. Our actions are now listen a_0, go left a_1 and go right a_2.

Transitions

a_0	s_0	s_1	s_2
s_0	1	0	0
s_1	0	1	0
s_2	0	0	1

a_1	s_0	s_1	s_2
s_0	0.1	0	0.9
s_1	0	0.9	0.1
s_2	0	0	1

a_2	s_0	s_1	s_2
s_0	0.9	0	0.1
s_1	0	0.1	0.9
s_2	0	0	1

ii) z represents an observation that the bunny makes. z_0 is hearing a tiger in the right-hand room. z_1 is hearing a tiger in the left-hand room.

$$p(z_0|s_0) = 0.8$$
$$p(z_0|s_1) = 0.2$$
$$p(z_1|s_0) = 0.2$$
$$p(z_1|s_1) = 0.8$$

iii) Now that we have taken action a_0 and observed z_1 We can update our posterior belief, b. Through the equation:

$$b'(s') = \eta p(z|s'a) \sum_{s \in S} p(s'|s,a)b(s)$$

Where η is a normalising factor equal to:

$$\eta = \frac{1}{\sum_{s' \in S} p(z|s'a) \sum_{s \in S} p(s'|s,a)b(s)}$$

For this MDP a_0 has a probability of 1 to transition to the same state, i.e. when $s = s'$ then $p(s'|s,a_0)$ =1. So clearly when $s \neq s'$ then $p(s'|s,a_0) = 0$. Therefore:

$$\sum_{s \in S} p(s'|s,a)b(s) = b(s')$$

$$\eta = \frac{1}{\sum_{s' \in S} p(z|s'a)b(s')}$$

$$b'(s') = \frac{p(z|s'a)b(s')}{\sum_{s' \in S} p(z|s'a)b(s')}$$

So for this question and MDP (ignoring the absorbing state as $p(z|s_2a_0)$=0):

$$b'(s') = \frac{p(z_1|s'a_0)b(s')}{\sum_{s' \in S} p(z_1|s'a_0)b(s')}$$

$$b'(s') = \frac{p(z_1|s'a_0)b(s')}{p(z_1|s_0a_0)b(s_0) + p(z_1|s_1a_0)b(s_1)}$$

$$p(z_1|s_0a_0)b(s_0) + p(z_1|s_1a_0)b(s_1) = 0.2 \times 0.5 + \ 0.8 \times 0.5 = 0.5$$

Now we can work out the new posterior belief by going over each possible state:

$$b'(s_0) = \frac{p(z_1|s_0a_0)b(s_0)}{0.5}$$

$$b'(s_0) = \frac{0.2 \times 0.5}{0.5}$$

$$b'(s_0) = 0.2$$

$$b'(s_1) = \frac{p(z_1|s_1a_0)b(s_1)}{0.5}$$

$$b'(s_1) = \frac{0.8 \times 0.5}{0.5}$$

$$b'(s_1) = 0.8$$

c)i)Non-episodic tasks have no absorbing states, so will have no final timestep and they will always be able to make actions that move to a new state and/or generate a reward. Episodic tasks have absorbing states from which every action produces no reward and returns to the same state and so will have a final timestep in each episode when an absorbing state is reached.

ii) The discount factor for non-episodic tasks is there to prevent the rewards summing to infinity and to give more immediate rewards a greater importance. The discount factor for episodic tasks is unnecessary but can be used to give more immediate rewards greater importance.