Exercise Sheet - MDPs and Online Planning

Exercise 1 (policies)**.** A deterministic policy is defined as a mapping from states to action, that is $\pi(s)$ maps states to actions.

Assume an environment with 4 states (a, b, c, and d) and 4 possible actions (up, down, left ,right). Give a complete deterministic policy that represents the strategy depicted in the figure below

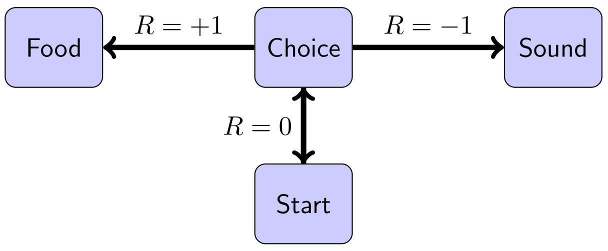
a $\rightarrow$	b $\leftarrow$
c $\uparrow$	d $\uparrow$

Solution:

1. $\pi(a)$ = left
2. $\pi(b)$ = right
3. $\pi(c)$ = up
4. $\pi(d)$ = up

▲

Exercise 2. A mouse-agent is situated in an environment with four states and four possible actions (up, down, left, right). The state transitions that are available in each state and their associated rewards are given by the arrows in the following figure:



Assume state transitions are deterministic, discount $\gamma = 0.5$ and we use the following value propagation function:

$$V(s) \leftarrow \max_a \left(\sum_{s'} T(s,a,s') \times [R(s,a,s') + \gamma V(s')] \right)$$

1. Give the values of s, a, s' for which $T(s,a,s') \neq 0$ and for which $R(s,a,s') \neq 0$
2. Iteratively calculate the V(s) values for all states until they have converged, start from V(s)=0.

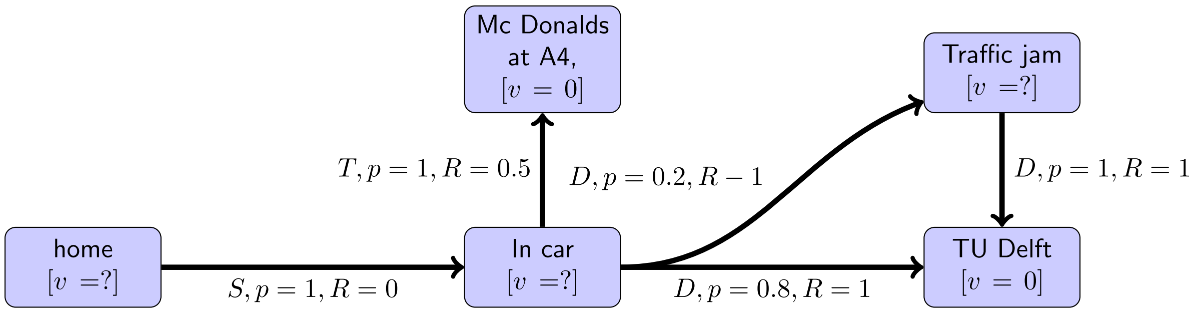
Solution:

1. Choice, left, Food and Choice, right, Sound
2. V(sound) = V(food) = 0 - will not change as these are final states.
   - V(start, choice) = (0,0)
   - V(start, choice) = (0,max(0,1,-1)) = (0,1)

- V(start, choice) = (0.5, max(0,1,-1)) = (0.5,1)
- V(start, choice) = (0.5, max(0.25,1,-1)) = (0.5,1)

▲

Exercise 3. Now lets assume a world in which actions are not deterministic. There are 5 states and 3 possible actions: S (Start), D (Drive), T (Turn). Each action a in state s has a probability p of ending up in state s' and with a reward R. All possible state transitions are given by arrows the figure below together with their probability and rewards. Note that action "drive" in state "in car" can lead either to "traffic jam" (0.2 chance) , or to "TU Delft" (0.8 chance).



Also we assume a "random policy" in states where multiple actions are available, and we assume $\gamma = 0.9$.

$$V^{\pi}(s) = \sum_{a} \pi(s,a) \sum_{s'} \mathcal{P}^{a}_{ss'} \left[\mathcal{R}^{a}_{ss'} + \gamma V^{\pi}(s')\right]$$

1. Use the Bellman equation (above) to calculate unknown $V(s)$. Hint: don't use iterations, instead first calculate values for which you know everything already and use those in the next calculation.
2. Now we know all the values we can find an improved policy. Calculate π(In Car), i.e. the best action when in car, using following formula:

$$\pi(s) = \text{argmax}_a \sum_{s'} \mathcal{P}^{a}_{ss'} \left[\mathcal{R}^{a}_{ss'} + \gamma V^{\pi}(s')\right]$$

Solution:

1. End states:
   V(TU Delft) = 0
   V(McDonalds) = 0
   Then:
   V(Traffic Jam) = 1
   Then:
   V(In car) = 0.5*0.5 (McDonalds) + 0.5*(0.8*(1 + 0.9*0) + 0.2*(-1+0.9*1)) (TU Delft, Traffic jam) = 0.25 + 0.4 - 0.01 = 0.64
   V(home) = 0.9*0.64 = 0.576
2. 0.5 (Turn) vs 0.8*(1 + 0.9*0) + 0.2*(-1+0.9*1) (Drive)
   0.5 vs 0.8-0.02 = 0.78
   Answer: Drive

▲

Exercise 4. For MCTS to converge to the optimal solution we can use a random rollout policy. In practice, a rollout policy that incorporates domain knowledge is often better than a random one.

1. Why does a rollout policy based on domain knowledge often improve the results of MCTS?

2. What is potentially a downside of using a rollout policy based on domain knowledge?

Solution:

1. In general, a rollout policy based on domain knowledge will be able to reduce the variance of simulation results, giving more accurate estimates of the value of the sampled trajectory.
2. Compared to using a random rollout policy, the rollout using domain knowledge may take considerable more time per rollout. It might include some computations or lookups in order to determine which action to take.

▲

Exercise 5 (MCTS for coordination in multiagent systems)**.** In Multiagent MDPs (MMDP) we are dealing with a joined actions space, resulting in a high branching factor when we want to apply MCTS. Consider a problem with n agents, where each agent has k actions.

1. What is the number of actions (i.e. the number of joint actions) that we have in the MMDP?

Another potential issue is that the number of states in a problem may increase a lot. Consider a 10 x 10 grid world and an agent that can walk around in this world.

2. How many possible states are there? What if we have 2 agents in this world?
3. What is an example problem where adding an additional agent would not result in a large increase in the number of states?

Solution:

1. k^n
2. 100
   If 2 agents can't occupy the same position: $\frac{100!}{(100-2)!} = 9900$.
3. An example could be a problem where agents have no physical presence (or a static physical presence, e.g. their location doesn't change). For instance if our state is a traffic block with 4 intersections, and we consider an intersection with traffic lights an agent, adding traffic lights to an intersection (that previously had no traffic lights) increases our number of agents by 1, without a large increase in the number of states.

▲