CS533
Intelligent Agents and Decision Making
MDP Basics and Finite Horizon Problems

1. Construct a simple Markov Decision Process such that the optimal policy for maximizing finite-horizon total reward must be non-stationary. That is, for some specified horizon, the MDP should not have a stationary policy that maximizes the finite horizon total reward.

2. In many problems, not all actions are applicable in all states and many actions only lead to a small number of next states, compared to the total number of states. In this question, we consider how the complexity of finite-horizon value iteration and policy evaluation can be improved for such problems.

   To capture the notion of applicable actions, suppose that we have a function LEGAL(s) that takes a state s and returns the set of legal actions in s. Also suppose that we have a function NEXT(s, a), which takes a state s and action a as input and returns the set of states that have non-zero probability of occurring after taking a in state s. That is,

   $$\text{NEXT}(s, a) = \{s' \mid T(s, a, s') > 0\}.$$

   Assume that we are considering an MDP with n states and m actions such that for any state s and action a we have LEGAL(s) $\leq k$ and NEXT(s, a) $\leq r$. Assume that the time and space complexity of evaluating the functions NEXT and LEGAL are linear in the sizes of their output (i.e. the number of elements in their sets).

   (a) Describe how to modify the finite-horizon policy evaluation algorithm described in class, using one or both of the new functions, so that the time complexity is improved when $r < n$ and $k < m$. What is the time complexity? The time complexity should be expressed in terms of r and k when possible and may also involve n and m.

   (b) Repeat part (a) but for the finite-horizon value iteration algorithm described in class.

3. Our basic definition of an MDP in class defined the reward function $R(s, a)$ to be a function of the state and action, which we will call an *sa-reward function*. It is also common to define a reward function to be a function of the state s, the action a, and resulting state s', written as $R(s, a, s')$, which we will call a *sas-reward function*. The meaning is that the agent gets a reward of $R(s, a, s')$ when it enters state s' after taking action a in state s. While this may seem to be a significant difference, it does not fundamentally reduce the modeling power, nor does it fundamentally change the algorithms that we have developed.

   (a) Give an example of a problem that is more naturally modelled using a sas-reward function compared to using an sa-reward function.

   (b) Modify the finite-horizon value iteration algorithm so that it works for sas-reward functions. Do this by writing out the new update equation that is used each iteration and explaining the modification from the equation given in class for sa-rewards.

   (c) It turns out that for value functions defined in terms of expected reward, we can always create an equivalent sa-reward MDP for any sas-reward MDP. That is, for an sas-reward MDP M we can create an sa-reward MDP M' such that the value function for any policy π in M is equivalent to the value function in M'. This means that solving M' is equivalent to solving M. Describe how to perform this transformation from M to M'.

4. It is also common to define MDPs using state reward functions, or s-reward functions, where the reward function $R(s)$ gives the reward for being in state s. It turns out that nothing fundamental is lost by moving from sa-rewards to s-rewards. Show how any MDP with an sa-reward function $R(s, a)$ can be transformed into a different, but "equivalent" MDP with an s-reward function $R(s)$. Here by equivalent, we mean that an optimal policy in the new MDP can be mapped to an optimal policy in the original MDP. Here we are considering the finite-horizon expected total reward value function. *Hint: It will be necessary for the new MDP to introduce new "book keeping" states that are not in the original MDP.*

5. **(k-th order MDPs.)** A standard MDP is described by a set of states S, a set of actions A, a transition function T, and a reward function R. Where $T(s, a, s')$ gives the probability of transitioning to s' after taking action a in state s, and $R(s, a)$ gives the immediate reward of being in state s and taking action a.

   A k-order MDP is described in the same way with one exception. The transition function T depends on the current state s and also the previous $k-1$ states. That is, $T(s_{k-1}, \ldots, s_1, s, a, s') = \Pr(s'|a, s, s_1, \ldots, s_{k-1})$ gives the probability of transitioning to state s' given that action a was taken in state s and the previous $k-1$ states were $(s_{k-1}, \ldots, s_1)$.

   Given a k-order MDP $M = (S, A, T, R)$ describe how to construct a standard (first-order) MDP $M' = (S', A', T', R')$ that is equivalent to M. Here equivalent means that a solution to M' can be easily converted into a solution to M. Be sure to describe S', A', T', and R'. Give a brief justification for your construction.

6. Suppose that in a finite-horizon setting, we would like the reward function to depend on the time-to-go. That is, the reward function will be of the form $R(s, a, t)$, which says that we get reward $R(s, a, t)$ for being in state s when the time-to-go is t and taking action a. Can finite-horizon value iteration be modified to take this reward function into account? If so, show how to modify the equations. If not, then give an argument why.