AI 533 **Intelligent Agents and Decision Making** Winter 2025

OREGON STATE UNIVERSITY

School of Electrical Engineering and Computer Science

Instructor: Sandhya Saisubramanian

Total points: 100 Assignment 1: MDP Due date: Jan 27, 2025

Instructions: Collaboration is not allowed on any part of this assignment. It is acceptable to discuss concepts or clarify questions but you must document who you worked with for this assignment. Copying answers or reusing solutions from individuals/the internet is unacceptable and will be dealt with strictly. Solutions must be typed (hand written and scanned submissions will not be accepted) and saved as a .pdf file.

1. **(15 points)** MDP design. For the following example scenarios, you must (1) write a factored state representation, discuss if it is discrete or continuous, and whether your state representation satisfies the Markov property; (2) *list* the actions (e.g. move up, move right, etc.) the agent should have to accomplish the assigned task; (3) write a reward function such that optimizing the reward function will help the agent accomplish its task efficiently (optimally) and briefly explain your answer. You are free to choose $R(s)$, $R(s,a)$ or $R(s,a,s')$ for your reward function. You can write your reward function mathematically or describe it clearly in text. Proposed reward functions must be reasonable, practical, and must not lead to unsafe behavior.

   (i) A robotic vacuum cleaner is assigned the task of removing dirt from the floor

   (1) Factored state representation:

   $< robot\ pos(x,y),\ CheckDirt($*0:clean, 1:dirty*$),\ CheckAllDirt($*0:No dirt, 1:there is dirt somewhere*$) >$

   It is in discrete state space since the cells in the grid world and whether dirt is removed or not are not continuously changing while tasking. For example, a specific location of the cell without dirt, which can be expressed as (1,0,0,1), has a fixed value. Moreover, the robotic vacuum cleaner only needs to check the dirt at the current position for cleaning without knowing the history of cleaning dirt, which means that it satisfies the Markov property.

   (2) Action set: $< MoveUp, MoveDown, MoveLeft, MoveRight >$

   (3) Reward for agent's tasks:

$$R(s,a) = \begin{cases} +100 & \textit{if there is no dirt on the floor, terminate the process} \\ +10 & \textit{if the robot finds dirt and cleans a dirty cell} \\ -1 & \textit{if the robot steps} \end{cases}$$

   To remove dirt from the floor, which is a grid world, a robot vacuum cleaner needs to know its position and need to check whether the current cell has dirt or not. For encouraging efficient navigation, a reward for cleaning is +20 when the robot successfully removes dirt in the cell, a reward for finishing tasks is +50 when removing all dirt from the floor, and a penalty for each movement is -1. The robot also needs information on the terminal state in which all cells are cleaned since it does not know where the dirt is somewhere else.

   (ii) A legged robot wants to run a marathon and reach the finish line as quickly as possible.

   (1) Factored state representation: $< robot\ pos : (x,y),\ direction,\ velocity >$

Features in the factored state representation are continuous state since these can be changed continuously in real-world. For example, to reach the goal, the robot adjusts its position, direction, and velocity when entering a curve, avoiding out of the marathon path. When it comes to the robot's decision making, it produces the next decision based on the current state (not the whole trajectory), satisfying the Markov property.

(2) Action set: $< Move\ forward,\ ChangeDirection,\ speed\ up,\ speed\ down >$

(3) Reward for achieving agent's task:

$$R(s,a) = \begin{cases} +100 & \text{if the robot attains the goal} \\ +30 & \text{if the robot is on the marathon path safely} \\ -10 & \text{if the robot is out of the marathon course} \\ -1 & \text{every at certain times} \end{cases}$$

The objective of the reward function focuses on how the legged robot reaches the target location, goal. To make the robot hurry, a penalty of -1 is applied at a certain time like every 5 seconds (it can be varied). There is a possibility that the robot is out of the lane, so it also yields a penalty of -10 whereas it will receive a reward of +30 while walking or running if the robot operates well and safely on the marathon course. Finally, when the robot achieves its goal, a reward for task completion of +100.

(iii) An autonomous car whose decisions must minimize the mean commute for all drivers (those driven by humans and those driven by AI)

(1) Factored state representation:

$< car\ pos(x,y),\ T_i(Driver\ i's\ commute\ time),\ C_j(Road\ j\ congestion) >$

Car position, commute time of all drivers, and road congestion are continuous states, depending on which road the autonomous car chooses. The decisions the car can make are based on the current state like its position, all other drivers' commute time, and the number of cars on the j road, not the choices the car made long ago. Thus, it satisfies the Markov property.

(2) Action set: $< ChangeRoute_j, Speed\ up, Speed\ down >$

(3) Reward for achieving agent's task:

$R(s,a) = +20 \cdot \text{(Reduced MeanCommuteTime)} - 10 \cdot \text{(safety issue)} - 5 \cdot \text{(Increase congestion)}$

Whenever the car chooses the way to reduce the mean commute for all drivers, a reward of +20 with the amount of reduced time. However, a penalty of -10 with safety issue can be applied when the car violates the speed limit or drives slowly, and a penalty of -5 with an increase in congestion on a certain road, which can possibly make other drivers' commute time delayed.

(iv) An underwater robot that must monitor the health of corals

(1) Factored state representation:

$< robot's\ pos(x,y,z),\ CheckCoralsHealth(healthy,\ sick,\ dead),\ robot's\ battery(\%) >$

Underwater robot's position and battery is continuous state because they are not fixed state, depending on the situation such as exploration and return to the base station for recharging. However, the CheckCoralsHealth state is discrete since there are only 3 fixed categories, healthy , sick, and dead. These features can be affected by the current state. For example, if the robot battery, which can serve as exploration time, almost runs out (near 10%), the robot will need to return to the base station. However, if not, the robot will explore and check the health of the corals. Therefore, it follows the Markov property.

(2) Action set:

$< Forward,\ Backward,\ MoveLeft,\ MoveRight,\ MoveUp,\ MoveDown,\ Scan,\ Recharge >$

(3) Reward for achieving agent's task:

$R(s,a) = +30\cdot$(Number of Successfully Scanning corals)$+5\cdot$(Unscanned area coverage)$-0.1\cdot$(consumed battery)

The objective of the reward function is to make an underwater robot scan as many corals as possible and cover new areas. If the robot successfully scans the number of corals, a reward of +30. For expanding coverage, a reward of +5. A penalty of -0.1 is a constraint to encourage further efficient navigation.

(v) An autonomous robot that is tasked with irrigating and fertilizing the crops to maximize crop yield, without adversely affecting crop and soil health.

(1) Factored state representation:

$< Robot's\ pos(x,y),\ Remaining\ water(\%), fertilizer(\%),\ Crop_i's\ Moisture(\%),$
$Nutrient(\%),\ Height(feet),\ Soil's\ pH(0.-14.) >$

All features in the factored state representation are continuous state since they are not fixed value, meaning that they can be changed continuously while irrigating and fertilizing the crops. These are the necessary information about robot, crops, and soil at the current step. The next state depends only on the current state and the action taken. Hence, the state representation satisfies the Markov property.

(2) Action set:

$< Irrigate,\ Fertilize,\ Refill,\ MoveUp,\ MoveDown,\ MoveLeft,\ MoveRight,\ ScanCrop,\ ScanSoil >$

(3) Reward for achieving agent's task:

$R(s,a) = +30\cdot(Crop_i's\ Height(\geq 4feet),\ Soil's\ pH(near\ 7)) - 5\cdot$ *(OverIrrigation&Overfertilization)*
$-\ 2\cdot$ *(Soil degradation)* $-\ 0.5\cdot$ *(Resource Usage)*

To maximize crops yield, a reward for Crop's Height (greater than or equal to 4 feet) and Soil's pH (near 7) of +30. For encouraging resource efficiency and preserving soil, the robot should avoid excessive usage of water and fertilizer and care of soil degradation, so a penalty for overirrigation and overfertilization of -5, a penalty for excessive usage of -0.5, and a penalty for soil degradation of -2.

2. **(15 points)** Given an MDP $M = (S, A, T, R, \gamma)$ with a fixed state state s_0 and a fixed policy π, the probability that the action at time $t = 0$ is $a \in A$ is:

$$\Pr(A_0 = a) = \pi(s_0, a).$$

Similarly, the probability that the state at time $t = 1$ is $s \in S$ is:

$$\Pr(S_1 = s) = \sum_{a_0\in A} \pi(s_0, a_0)T(s_0, a_0, s).$$

Write a similar mathematical expression (using only S, A, T, R, γ, π and Bayes' theorem) for the following:

(i) The expected reward at time $t = 6$ given that the action at time $t = 3$ is $a \in A$ and the state at time $t = 5$ is $s \in S$. Use $R(s, a)$ for reward notation.

$$E[R_6|A_3 = a,\ S_5 = s] = \textstyle\sum_{s_3\in S} \pi(s_3, a) \sum_{s_4\in S} T(s_3, a, s_4) \sum_{a_4\in A} \pi(s_4, a_4)T(s_4, a_4, s)$$
$$\textstyle\sum_{a_5\in A} \pi(s, a_5) \sum_{s_6\in S} T(s, a_5, s_6) \sum_{a_6\in A} \pi(s_6, a_6)R(s_6, a_6)$$

(ii) The probability that the action at time $t = 16$ is $a' \in A$ given that the action at time $t = 15$ is $a \in A$ and the state at time $t = 14$ is $s \in S$.

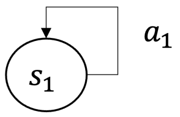
$$Pr(A_{16} = a'|A_{15} = a, S_{14} = s) = \textstyle\sum_{a_{14}\in A} \pi(s, a_{14}) \sum_{s_{15}\in S} T(s, a_{14}, s_{15})\pi(s_{15}, a)$$
$$\textstyle\sum_{s_{16}\in S} T(S_{15}, a, s_{16})\pi(s_{16}, a')$$

3. **(5 points)** How many deterministic policies (optimal or otherwise) exist for an MDP with 5 states and 10 actions?

A deterministic policy maps each state to exactly one action, so one action out of 10 available actions can be selected for each of the 5 states. Thus, the total number of policies is:

$$10 \times 10 \times 10 \times 10 \times 10 = 10^5 = 100,000$$

4. **(10 points)** For the MDP in the following figure with one state and one action, let $R(s_1) = 0$ and $V_0(s_1) = 5$. (i) Will value iteration converge when $\gamma = 1$? Briefly explain your answer. (ii) Will value iteration converge when $\gamma = 0.9$? Briefly explain your answer.



(i): The value iteration will converge when $\gamma = 1$ since the value function V has the same value in each iteration.

$$V_{k+1}(s_1) = R(s_1) + \gamma \cdot T(s_1, a_1, s_1) \cdot V_k(s_1)$$
$$V_{k+1}(s_1) = 0 + 1 \cdot 1 \cdot V_k(s_1)$$
$$V_{k+1}(s_1) = V_k(s_1)$$

As can be seen above, the value remains constant at 5, meaning that it does not converge to a unique fixed point. Thus, it is not converge.

(ii): The value iteration will not converge when $\gamma = 0.9$ since for every iteration the value gradually decreases.

$$V_{k+1}(s_1) = R(s_1) + \gamma \cdot T(s_1, a_1, s_1) \cdot V_k(s_1)$$
$$V_{k+1}(s_1) = 0 + 0.9 \cdot 1 \cdot V_k(s_1)$$
$$V_{k+1}(s_1) = 0.9 \cdot V_k(s_1)$$

If the value of k goes infinite, the sequence is decreasing and approaching 0, value iteration converges to 0.

5. **(15 points)** Prove the following two statements mathematically or provide an example to demonstrate the property.
Statement 1: Multiplying all rewards (of a finite, discrete MDP with bounded rewards) by a positive scalar does not change the optimal policy.
Statement 2: Adding a positive constant to all rewards (all states or state-action pairs or state-action-successor pairs in the MDP) of a finite MDP with bounded rewards changes the optimal policy.

Statement 1)

All rewards multiplying by a positive scalar $c > 0$ becomes:

$$R'(s, a) = cR(s, a)$$

Then, a new value function becomes:

$$V'^{\pi}(s) = E\left[\sum_{t=0}^{\infty} \gamma^t cR(s, \pi(s))|\pi,\ S_0 = s\right]$$
$$= cE\left[\sum_{t=0}^{\infty} \gamma^t R(s, \pi(s))|\pi,\ S_0 = s\right]$$
$$= cV^{\pi}(s)$$

Since the value function scales linearly with the scalar c, the Bellman optimality equation would also be:

$$V'^{*}(s) = max_{a\in A}\left[cR(s,a) + \gamma \sum_{s'\in S} T(s,a,s')V'^{*}(s')\right]$$
$$= c\, max_{a\in A}\left[R(s,a) + \gamma \sum_{s'\in S} T(s,a,s')V'^{*}(s')\right]$$
$$= cV^{*}(s)$$

The optimal policy is determined by the action that maximizes the value:

$$\pi^{*} \in argmax_{\pi\in\prod} E\left[\sum_{t=0}^{T} R_t|\pi\right]$$

Thus, multiplying by a positive scalar does not affect the optimal policy.

Statement 2)

Adding a positive constant c to all rewards is:

$$R'(s,a) = R(s,a) + c$$

Then, a new value function under policy π becomes:

$$V'^{\pi}(s) = E\left[\sum_{t=0}^{\infty} \gamma^t (R(s, \pi(s)) + c)|\pi,\ S_0 = s\right]$$
$$= E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, \pi(s))|\pi, S_0 = s\right] + E\left[\sum_{t=0}^{\infty} \gamma^t c|\pi, S_0 = s\right]$$

Based on the infinite geometric series, it would be:

$$= V^{\pi}(s) + \frac{c}{1-\gamma}$$

The value function is shifted by a constant $\frac{c}{1-\gamma}$. The Bellman equation for the optimal value function becomes:

$$V'^*(s) = max_{a \in A} \left[R(s,a) + c + \gamma \sum_{s' \in S} T(s,a,s') V'^*(s') \right]$$

$$= max_{a \in A} \left[R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') V^*(s') \right] + \frac{c}{1-\gamma}$$

Because of the constant term above, the ranking of actions may change, depending on how sensitive the rewards are. If all actions previously had the same expected value, adding a positive constant can skew the agent's decisions toward shorter-term rewards. Thus, it potentially changes the optimal policy.

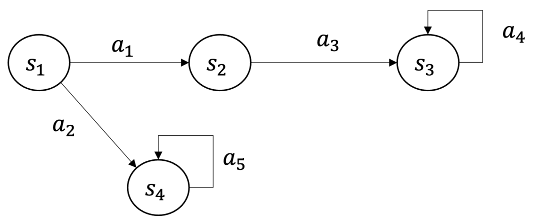
6. **(30 points)** In this question, the goal is to demonstrate that value improvement in value iteration is not monotonic. While the values and the policy converge to optimal at termination, for a finite, bounded MDP, the values (and the resulting policy) before convergence of value iteration are not guaranteed to improve in a monotonic manner.

   Consider the MDP in Figure 1 with four states, five actions that have deterministic transitions, and discount factor $\gamma = 1$. The reward for being in each state, $R(s)$, is reported in the table in Figure 1. Let V_i and V_{i+1} denote value functions from two iterations of value iteration on this problem **before convergence**. Let π_i and π_{i+1} denote the policies that are greedy with respect to these value functions.

   (i) You are required to assign an initial value (V_0) to each of the state such that the expected reward following the greedy policy is not monotonic, $\pi_{i+1} < \pi_i$. Initialize V_0 such that at some finite iteration count i before convergence of VI, the expected reward following π_{i+1} is less than the expected reward following π_i, causing a change in the *current* optimal action at least in one state.

   For example, let a_1 and a_2 denote the actions available in state s. Let the true optimal action in s be $a^* = a_1$. You must initialize V_0 such that at iteration i, the policy that is greedy on the values $\pi(s) = a_1$ but at iteration $i+1$, $\pi(s) = a_2$ and in some $i+k$, the policy stabilizes to the true optimal policy $\pi(s) = a_1$.

   (ii) Show your calculation of value iteration to support (i).



State	Reward R(s)
s_1	0
s_2	-1
s_3	+2
s_4	0

Figure 1: Graph for analyzing monotonicity of value iteration

An initial value (V_0) to each of the state is important for inducing the agent to choose greedy action at iteration i, another possible action at iteration $i+1$, and finally stabilize to the true optimal policy. So, the initial values of the value function can be assigned:

$$V_0(s_1) = 0, \; V_0(s_2) = 5, \; V_0(s_3) = 2, \; V_0(s_4) = 3$$

The value function $V_{t+1}(s)$ equation is:

$$V_{t+1}(s) = max_a \big[R(s,a) + \gamma \sum_{s' \in S} T(s,a,s')V_t(s)\big]$$

The discount factor $\gamma = 1$, and the transition function $T(s,a,s') = 1$ since there are no fail cases for each state. This simplify the equation:

$$V_{t+1}(s) = max_a \big[R(s,a) + V_t(s)\big]$$

Due to the assigned values in value function at iteration 0, the value improvement in value iteration at certain time step is not monotonic. The next values of each state are:

$$t+1=1) \quad \begin{cases} s_1 : \begin{cases} a_1 \rightarrow 0+5 =< 5 > \\ a_2 \rightarrow 0+3 = 3 \end{cases} & \Rightarrow \pi(s_1) = a_1 \\ s_2 : a_3 \rightarrow -1+2 = 1 \\ s_3 : a_4 \rightarrow 2+2 = 1 \\ s_4 : a_5 \rightarrow 0+3 = 3 \end{cases}$$

At iteration 1, the value function $V_1(s)$ is $V_1(s_1) = 5,\ V_1(s_2) = 1,\ V_1(s_3) = 4,\ V_1(s_4) = 3$

Likely, at iteration 2:

$$t+1=2) \quad \begin{cases} s_1 : \begin{cases} a_1 \rightarrow 0+1 = 1 \\ a_2 \rightarrow 0+3 =< 3 > \end{cases} & \Rightarrow \pi(s_1) = a_2 \\ s_2 : a_3 \rightarrow -1+4 = 3 \\ s_3 : a_4 \rightarrow 2+4 = 6 \\ s_4 : a_5 \rightarrow 0+3 = 3 \end{cases}$$

The value function $V_2(s)$ is $V_2(s_1) = 3,\ V_2(s_2) = 3,\ V_2(s_3) = 6,\ V_2(s_4) = 3$

As can be seen above, unlike the policy $\pi(s) = a_1$ at iteration 1, action a_2, which is suboptimal, is chosen, meaning that value improvement in value iteration is not monotonic.

Then, at iteration 3:

$$t+1=3) \quad \begin{cases} s_1 : \begin{cases} a_1 \rightarrow 0+3 = 3 \\ a_2 \rightarrow 0+3 = 3 \end{cases} & \Rightarrow \pi(s_1) = a_1 \ or \ a_2 \\ s_2 : a_3 \rightarrow -1+6 = 5 \\ s_3 : a_4 \rightarrow 2+6 = 8 \\ s_4 : a_5 \rightarrow 0+3 = 3 \end{cases}$$

The value function $V_3(s)$ is $V_3(s_1) = 3,\ V_3(s_2) = 5,\ V_3(s_3) = 8,\ V_3(s_4) = 3$

At iteration 4:

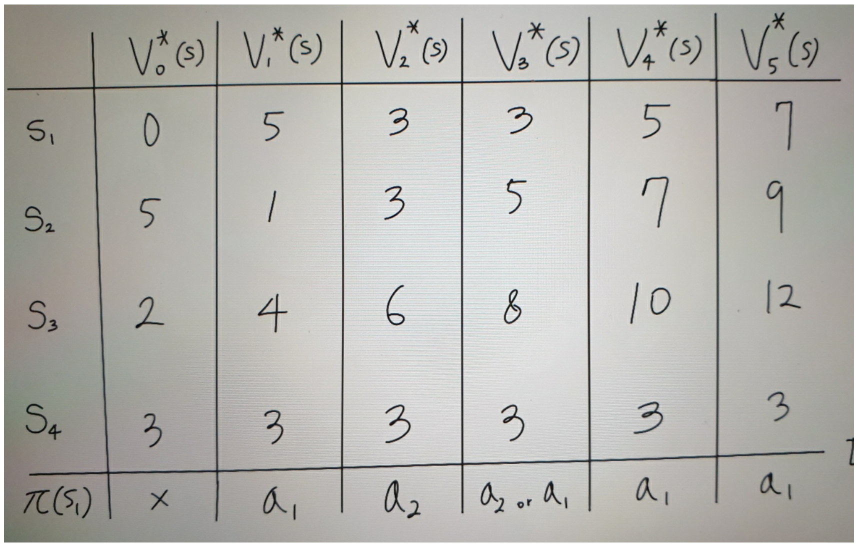
$$t+1=4) \quad \begin{cases} s_1 : \begin{cases} a_1 \rightarrow 0+5 =< 5 > \\ a_2 \rightarrow 0+3 = 3 \end{cases} & \Rightarrow \pi(s_1) = a_1 \\ s_2 : a_3 \rightarrow -1+8 = 7 \\ s_3 : a_4 \rightarrow 2+8 = 10 \\ s_4 : a_5 \rightarrow 0+3 = 3 \end{cases}$$

The value function $V_4(s)$ is $V_4(s_1) = 5,\ V_4(s_2) = 7,\ V_4(s_3) = 10,\ V_4(s_4) = 3$

At iteration 5:

$$t+1=5)\quad \begin{cases} s_1 : \begin{cases} a_1 \to 0+7 =< 7 > \\ a_2 \to 0+3 = 3 \end{cases} & \Rightarrow \pi(s_1) = a_1 \\ s_2 : a_3 \to -1+10 = 9 \\ s_3 : a_4 \to 2+10 = 12 \\ s_4 : a_5 \to 0+3 = 3 \end{cases}$$

The value function $V_5(s)$ is $V_5(s_1) = 7,\ V_5(s_2) = 9,\ V_5(s_3) = 12,\ V_5(s_4) = 3$



	$V_0^*(s)$	$V_1^*(s)$	$V_2^*(s)$	$V_3^*(s)$	$V_4^*(s)$	$V_5^*(s)$
S_1	0	5	3	3	5	7
S_2	5	1	3	5	7	9
S_3	2	4	6	8	10	12
S_4	3	3	3	3	3	3
$\pi(S_1)$	×	a_1	a_2	a_2 or a_1	a_1	a_1

Figure 2: Result table

7. **(10 points)** In class, we proved the contraction mapping for the Bellman equation, independent of the policy. You are required to prove that the Bellman backup operator for a particular policy converges. For a deterministic policy π and $0 \leq \gamma < 1$, let us define a contraction operator $(B^\pi V)(s) = R(s, \pi(s)) + \gamma \sum_{s'} T(s, \pi(s), s')V(s')$. Prove that $\|B^\pi V - B^\pi V'\| \leq \gamma \|V - V'\|$. Hint: use the max-norm operator $\|v\| = max_s|v(s)|$ and follow steps similar to the proof in lecture slides.

   By definition of the max-norm of V, it returns the maximum absolute value of the vector:

$$||V|| = max_s|V(s)|$$

   According to the L5_ExactMethods, $\|B^\pi V - B^\pi V'\|$ can be expanded:

$$\left\|B^\pi V - B^\pi V'\right\| = max_s|B[V(s)] - B[V'(s)]|$$

$$= max_s \left| \begin{matrix} max_{\pi(s)} R(s, \pi(s)) + \gamma \sum_{s'} T(s, \pi(s), s')V(s') - \\ max_{\pi(s)} R(s, \pi(s)) + \gamma \sum_{s'} T(s, \pi(s), s')V'(s') \end{matrix} \right|$$

With the property $|max_x f(x) - g(x)| \leq max_x |f(x) - g(x)|$,

$$||BV - BV'|| \leq max_s max_a \left| \begin{matrix} R(s,\pi(s)) + \gamma \sum_{s'} T(s,\pi(s),s')V(s') - \\ R(s,\pi(s)) + \gamma \sum_{s'} T(s,\pi(s),s')V'(s') \end{matrix} \right|$$

$$\leq max_x max_{\pi(s)} \gamma \left| \sum_{s'} T(s,\pi(s),s')(V(s) - V'(s)) \right|$$

Simplifying terms:

$$\leq max_s max_{\pi(s)} \gamma \left| max_{s'} (V(s) - V'(s)) \right|$$

Max over s', a are irrelevant at this point:

$$= \gamma \left| max_s (V(s) - V'(s)) \right|$$

$$= \gamma ||V - V'||$$

Thus, the inequality $\|B^\pi V - B^\pi V'\| \leq \gamma \|V - V'\|$ *holds.*