EE675A - Introduction to Reinforcement Learning,

Lecture 9: Intro to MDP and Bellman Expectation equations

08.02.2023

Lecturer: Subrahmanya Swamy Peruru *Scribe: Subhrojyoti Chatterjee, Ankit Jagdish Shinde*

In the previous lecture, we discussed about Policy Gradient Algorithm and Contextual Bandits. In this lecture, we will be discussing the full RL setting along with Markov Decision Processes, Value Function, Action Function and Bellman Expectations Equation

1 Full Reinforcement Learning Setting

In this section we will discuss about the basic terminologies used in Reinforcement Learning and a flow diagram of how the environment interacts with the agent in a reinforcement learning setting.

1.1 Basic Terminologies

Below is a list of the basic terminologies required to define a Reinforcement Learning problem -

- **Agent** - The entity that uses a policy to maximize the expected return gained from transitioning between states of the environment.
- **Environment** - The world that contains the agent and allows the agent to observe that world's state. For example, the represented world can be a game like chess or a physical world like a maze. When the agent applies an action to the environment, then the environment transitions between states.
- **State (S_t)** - The parameter values that describe the current configuration of the environment, which the agent uses to choose an action.
- **Action (A_t)** - The mechanism by which the agent transitions between states of the environment. The agent chooses the action by using a policy.
- **Reward (R_t)** - A feedback returned to the agent from the environment to evaluate the action of the agent.
- **Termination Condition** - The conditions that determine when an episode ends, such as when the agent reaches a certain state or exceeds a threshold number of state transitions. For example, in tic-tac-toe (also known as noughts and crosses), an episode terminates either when a player marks three consecutive spaces or when all spaces are marked.
- **Policy** - An agent's probabilistic mapping from states to actions.

- **Return** (G_t) - Given a certain policy and a certain state, the return is the sum of all rewards that the agent expects to receive when following the policy from the state to the end of the episode. The agent may account for the delayed nature of expected rewards by discounting rewards according to the state transitions required to obtain the reward.

1.2 Flow Diagram of Reinforcement Learning Setting

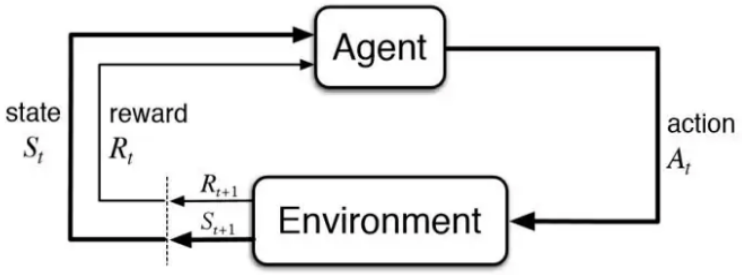

Figure 1: Flow Diagram of an RL Algorithm

2 Markov Decision Processes

In this section, we will firstly discuss about independent random variables and Markov Chains. Finally, we will discuss about Markov Decision Processes and their types.

2.1 Independent Random Variables and Markov Chains

Two random variables X and Y are independent if knowing the value of one of them does not change the probabilities for the other one. Intuitively extending the definition of independence to n discrete random variables $X = (X_1, ..., X_n)$, we have that X is a set of n independent random variables if $\forall i \in [1, n]$ knowing the values of $X_j \forall j \neq i$ does not tell us anything about the value of X_i. Mathematically one of the ways to express the condition for independence of n discrete random variables is :

$$P(X_i = x_i | X_j = x_j \ \forall j \in [1, n] \text{ and } j \neq i) = P(X_i = x_i) \ \forall x_1, x_2, ..., x_n$$

Markov Property: A stochastic process has the Markov property if the conditional probability distribution of future states of the process (conditional on both past and present values) depends only upon the present state; that is, given the present, the future does not depend on the past. Mathematically it can be defined as follows :

$$P(X_n = x_n | X_{n-1} = x_{n-1}, ..., X_1 = x_1, X_0 = x_0) = P(X_n = x_n | X_{n-1} = x_{n-1}) \ \forall x_1, x_2, ..., x_n$$

Markov Process: A Markov process is a stochastic process that satisfies the Markov property. Informally, this may be thought of as, "What happens next depends only on the state of affairs now."

Markov Chain: A Markov chain is a discrete-time process for which the future behaviour, given the past and the present, only depends on the present and not on the past. A Markov process is the continuous-time version of a Markov chain.

Transition Matrix: If a Markov chain consists of k states, the transition matrix is the k by k matrix (a table of numbers) whose entries record the probability of moving from each state to another state. The rows of the matrix correspond to the current state and columns correspond to the next state. For example, the entry at row 1 and column 2 records the probability of moving from state 1 to state 2.

Let us take an example to better understand how to build a transition matrix.
Problem: We flip a coin which has probability p of landing on heads. We keep a reward system where we get Rs.1 if the coin lands on heads and lose Rs.1 if the coin lands on tails. Let our random variable (X) be the amount of money(in Rs.) we have and X = 0 and X = 3 be the terminal states.

First we will verbally reason for why the problem satisfies the Markov Property. Let us consider that X = q. So we can clearly see that the next state can be X = q+1(if coin lands on heads) or X = q-1(if coin lands on tails). Thus probability of next state being X = q+1 is p and being X = q-1 is 1-p.

As we have probabilities of next state from the knowledge of the current state only, we can say that it is a Markov Chain. Using the above obtained result we will build our transition matrix(T) as well.

$$\text{T} = \begin{pmatrix} - & - & - & - \\ 1-p & 0 & p & 0 \\ 0 & 1-p & 0 & p \\ - & - & - & - \end{pmatrix}$$

The first and last rows are not defined as they are terminal states and if you are in them then the game is already over.

2.2 Markov Decision Processes

Definition: A graph representing the decision-making model where decisions (or actions) are taken to navigate a sequence of states under the assumption that the Markov property holds with a numerical reward obtained on transitioning between the states.

Requirements to characterize an MDP:

- **State Space**: The state space S is a set of all the states that the agent can transition to
- **Action Space**: The action space A is a set of all actions the agent can act out in a certain environment.
- **Policy**: The policy π is the probability of taking an action $a \in A$ when in state $s \in S$.
- **Dynamics of the MDP**: In a finite MDP, the sets of states, actions, and rewards (S, A, and R) all have a finite number of elements. In this case, the random variables R_t and S_t have well defined discrete probability distributions dependent only on the preceding state and action. That is, for particular values of these random variables, $s^{'} \in S$ and $r^{'} \in R$, there is a probability of those values occurring at time t, given particular values of the preceding state and action:

$$P(s^{'}, r^{'}|s, a) = P[S_t = s^{'}, R_t = r^{'}|S_{t-1} = s, A_{t-1} = a] \; \forall s^{'}, s \in S, r \in R \text{ and } a \in A$$

Types of Markov Decision Processes: There are two types of MDP's, namely Episodic MDP's and Continuous MDP's.

- **Episodic MDP**: These are Markov Decision Processes with well-defined terminal state. In these MDP's we intend to maximize the return

  $$G_t = \sum_{i=t+1}^{T} R_i$$

  The returns in these cases are finite.

- **Continuous MDP**: These are Markov Decision Processes that go on forever and do not have a terminal state. These MDP's do not have necessarily have a finite value of return and thus we use a discounting factor(γ) to make the return finite as well as to account for the delayed nature of expected rewards. Thus the expression for return in this case will be

  $$G_t = \sum_{i=t+1}^{\infty} \gamma^{i-t-1} R_i$$

  We can trivially observe that while using $\gamma = 0$, we only care about immediate reward and while using $\gamma = 1$ we care about long term rewards as well.

In the MAB problem, if given the values of true means of the arms i.e. $P(r|a)$ it becomes a trivial problem. However, in the MDP problem, despite being given the entire dynamic of the MDP i.e $P(s^{'}, r^{'}|s, a)$, it is not a trivial problem to obtain the optimal policy.

3 Value Function and Action Value Function

In this section we first define the value function and action value function and then express the value function in terms of the action value function.

3.1 Value Function

Denoted by $V_\pi(s)$, this value function measures potential future rewards we may get from being in this state s and following the policy π. Mathematically, we can define the value function as:

$$V_\pi(s) = E_\pi[G_t|S_t = s]$$

3.2 Action Value Function

Denoted by $Q_\pi(s, a)$, the action-value-function returns the value, i.e. the expected return for using action a in a certain state s while following a policy π from the next action. Mathematically, we can define the action-value function as:

$$Q_\pi(s, a) = E_\pi[G_t|S_t = s, A_t = a]$$

3.3 Relation between Value Function and Action Value Function

For a deterministic policy, we can clearly see that taking action a = $\pi(s)$ will mean that we follow policy π for state s as well. Thus,

$$V_\pi(s) = Q_\pi(s, \pi(s))$$

for deterministic policy π.

We can extend a similar idea to stochastic processes and say that the value function equals the sum of action value functions for all actions belonging to action space weighted by their probability of being taken in state s by policy π. Thus,

$$V_\pi(s) = \sum_{a \in A} \pi(a|s) Q_\pi(s, a)$$

4 Bellman's Expectation Equation

Firstly, let us talk about an important mathematical relation we will be used in deriving this relation.

$$E[X] = E_Y[E_X[X|Y]] \text{ —— Relation 1}$$

That is if you want to find the expected value of a random variable, you can first find the expected value of the variable conditioned over another and then find expected value over the other variable.

Let's now derive the Bellman Expectation Equations,

$V_\pi(s) = E_\pi[G_t|S_t = s]$
$V_\pi(s) = E_\pi[R_{t+1} + \gamma R_{t+2} + ...|S_t = s]$
$V_\pi(s) = E_\pi[R_{t+1} + \gamma G_{t+1}|S_t = s]$

Now applying the Relation 1

$V_\pi(s) = E_{a,r,s'}[E_\pi[R_{t+1} + \gamma G_{t+1}|S_t = s, S_{t+1} = s^{'}, R_{t+1} = r, A_t = a]]$
$V_\pi(s) = \sum_{a\in A,s'\in S,r}(P(a, r, s^{'}|s)E_\pi[R_{t+1} + \gamma G_{t+1}|S_t = s, S_{t+1} = s^{'}, R_{t+1} = r, A_t = a])$

Now,using Markov Chain Property and Sum of Expectations Property

$V_\pi(s) = \sum_{a\in A,s'\in S,r}(P(a, r, s^{'}|s)[r + \gamma E_\pi[G_{t+1}|S_{t+1} = s^{'}]])$
$V_\pi(s) = \sum_{a\in A,s'\in S,r}(P(a, r, s^{'}|s)[r + \gamma V_\pi(s^{'})]$

Thus we have a recursive set of equations in $V_\pi(s)$. If there are n states, there will be n equations and thus those n equations can be solved to get the solution for $V_\pi(s)$ for all states s.

The above set of equations are called Bellman Expectation Equations.

5 References

- **Chapter 3** of Reinforcement Learning-An Introduction by Richard S. Sutton and Andrew G. Barto.