# **MARKOV DECISION PROCESS**

Markov Decision Process (MDPs) are a classical formalization of sequential decision making, where actions influence not just immediate rewards, but also subsequent situations, or states, and through those future rewards. Thus, MDPs involve delayed reward and the need to trade off immediate and delayed reward. MDPs are a mathematically idealized form of the reinforcement learning problem for which precise theoretical statements can be made.

To understand MDP we first need to understand certain concepts, upon which the MDP is built.

## **AGENTS AND ENVIRONMENT**

MDPs can be regarded as the formal framing of the problem of learning through interactions to achieve a particular goal. The learner or the decision maker is known as the ***agent***. The agent is often something that decides what to do in a particular situation. The goal of every Reinforcement Learning problem is to teach the agent decisions to make the correct decisions in order to reach the goal.

Everything apart from the agent is known as the ***environment.*** It is the surroundings that interact with the agent. In a sense, the agent waits for a feedback from the environment, in order to decide. These interact continually, the agent selecting actions and the environment responding to these actions and presenting new situations to the agent.

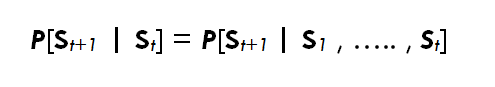
## **STATE, REWADRS AND ACTIONS**

The agent and environment, as described just now, interact at each of a sequence of discrete time steps, t = 0, 1, 2, 3, . . .. 2 At each time step t, the agent receives some representation of the environment’s ***State,*** and on that basis selects an ***Action.*** A state is something that is used to describe the current situation we are in. It describes the conditions of our environment. An action is the agent’s response to the current state and reward. The agent makes decisions in the form of actions performed. Because of the action, the agent receives a ***Reward*** from the environment. It is the environment’s response to the current action and state. The agent takes actions in order to maximize or minimize the total reward. In MDP, we can choose what kind of reward we favour through a discount factor Gamma. We can favour an immediate reward by keeping a high discount factor or also keep into account the later reward by keeping a low discount factor. More on this will be discussed later.

## **MARKOV PROPERTY**

We need to understand the Markov property before formulating MDP. Markov property states that

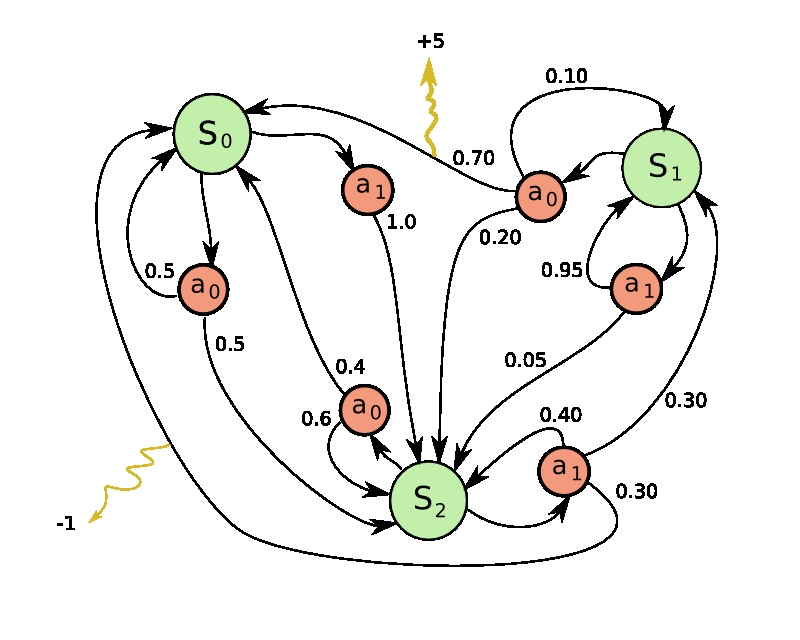
“***Future is independent of the past given the present”***

Various components of an MDP follow the markov property. It can mathematically be expressed as 

Moving from one state to another is known as ***Transition.*** The probability with which we move from one state to another is the transition probability. In the above formula, the markov property is applied to the transition probabilities. It can be interpreted as the probability of St+1 (state at time t+1 where current time is t) being the next state, only depends on the current state St and is independent of any other state in the past. Intuitively it can be said that the current state encompasses all the information of the past states. We can define state transition probability matrix P as P[i][j] tells us the probability of trasitioning from state i to state j.

## **MARKOV PROCESS OR MARKOV CHAINS**

A ***Markov proces****s* is a memory-less random process, i.e. a sequence of random states S1, S2, ….. with the Markov property. A Markov process or Markov chain is a tuple (***S***, ***P***) on state space ***S*** and transition function ***P***. When we sample a Markov chain from a MDP it is known as an episode.



In the above shown diagram an episode S0,S2,S1,S1….

The red circles indicate the actions that we can take while we are in a state. The numbers written on the arrows denote the probabilities.

More on this is discussed later.

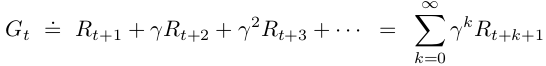
## **MARKOV REWARD PROCESS**

A Markov Reward Process or MRP is a quantifiable Markov process. That is, it has a value associated with itself. It tells us if we follow the Markov process, how much reward we will accumulate.

MRP can be described in the form of a tuple.  (***S***, ***P***, ***R***, **𝛾**) where ***S***is a finite state space, ***P***is the state transition probability function, ***R*** is a reward function, where,

***R****s* = 𝔼[***R****t+1* | ***S****t* = *S*],

It says how much immediate reward we expect to get from state *S*at the moment. The actual value produces by an MRP can be quantified using another quantity known as the return Gt. Which can be written as.



Where γ denotes the discount factor and γ is in the range [0,1]. It informs the agent of how much it should care about rewards now to rewards in the future. If (**γ**= 0), that means the agent is short-sighted, in other words, it only cares about the first reward. If (**γ** = 1), that means the agent is far-sighted, i.e. it cares about all future rewards. What we care about is the total rewards that we are going to get. Gamma is just a way for us to shift our emphasis on the type of reward we favour. It is trivial to mention that Gamma is problem specific and different values, might lead to different results.

Having a discount factor is important because it makes sure that the MDP converges. For example, suppose we have sampled a MRP from a state diagram with a cycle. Therefore we can sample an MRP whose length is infinite. If we do not use a discount factor, the return value of this MDP will blow up. Another reason for using a discount factor might be, that the future is uncertain. Hence we give less importance to the rewards we receive in future and more to what we receive now.

## **VALUE FUNCTION**

Almost all reinforcement learning algorithms involve estimating value functions—functions of states (or of state–action pairs) that estimate how good it is for the agent to be in a given state (or how good it is to perform a given action in a given state). The notion of “how good” here is defined in terms of future rewards that can be expected, or, to be precise, in terms of expected return.

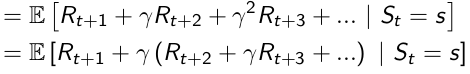


## **BELLMAN EQUATION**

Our goal was not just to create an MDP. Useful information can only be extracted from an MDP if it is solved. The agent tries to get the most expected sum of rewards from every state it lands in. In order to achieve that we must try to get the optimal value function, i.e. the maximum sum of cumulative rewards. This can be achieved by the Bellman Equation.

It decomposes the value function into 2 parts the immediate reward Rt+1 and the discounted value function γ\*V(St+1)­­­. It can be reached in the following steps.







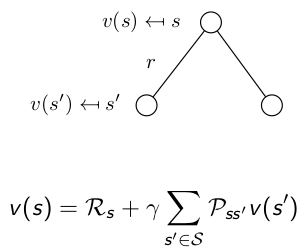
Since E(E(x)) is E(x), we can write this as



This gives us the ***Bellman Equation***

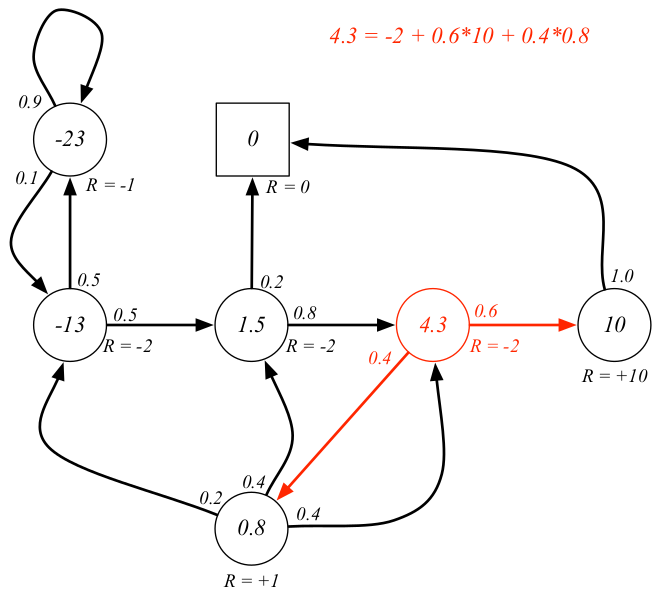


It can be better understood in a different form.



Where Pss’ is the Transition probability matrix. To develop the intuition for this, we can look at it in the following way. Suppose we are in a particular state. The value that we get from this state can be seen as the reward that we get from this state and the value we get if we move to other states from this state as we follow the markov chain. The transition probability matrix helps us capture the values of all the possible states that we can go to from this particular state, i.e.  the value of each possible successor state is multiplied by the probability that we land in it.

To further understand this, we can look at an example.



The value of the state in red is calculated as shown in the figure. The value of the state can be thought of as the reward in that state plus the value we get when we transition from that state which is 0.6 times the value of state on the left because our probability of going there is 0.6 and same way for the state at the bottom.

Before moving forward, we should know about one more concept.

## **POLICY**