**SCHOOL of AI Notes:**

**1. A Machine Learning journal by Luca Palmieri  
[Part0]** [**https://www.lpalmieri.com/posts/rl-introduction-00/#markov-decision-processes-mdps**](https://www.lpalmieri.com/posts/rl-introduction-00/#markov-decision-processes-mdps) **[Part1]** [**https://www.lpalmieri.com/posts/rl-introduction-01/#policies**](https://www.lpalmieri.com/posts/rl-introduction-01/#policies)

**A well-organized blog that helps people who study for the first time understand RL. It is suitable for the first  reading assignment with short length and interesting pictures attached.**

**2. Reinforcement Learning Demystified: Markov Decision Processes by Mohammad Ashraf  
[Part 1]** [**https://towardsdatascience.com/reinforcement-learning-demystified-markov-decision-processes-part-1-bf00dda41690**](https://towardsdatascience.com/reinforcement-learning-demystified-markov-decision-processes-part-1-bf00dda41690)

**This blog explained Richard Sutton's book to make it easy to understand, and it is a good starting point for those who want to study deeply in the future, using well-known lecture notes from David Silver.**

**3.  UCL Course on Reinforcement Learning by David Silver (2015)**[**http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching.html**](http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching.html)

**Lecture 1: Introduction to Reinforcement Learning**[**http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching\_files/intro\_RL.pdf**](http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching_files/intro_RL.pdf)

**Lecture 2: Markov Decision Processes**[**http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching\_files/MDP.pdf**](http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching_files/MDP.pdf)

**The famous David Silver's lecture on RL. Video and slide notes are quoted from most RL textbooks and lectures. Student MRP is already quoted in the blog above, and it's a popular example of MDP that everyone sees once in their course of study.**

**4. (bonus)  Reinforcement Learning: An Introduction by Richard S. Sutton and Andrew G. Barto (Second edition, 2018)**

[**https://drive.google.com/file/d/1xeUDVGWGUUv1-ccUMAZHJLej2C7aAFWY/view**](https://drive.google.com/file/d/1xeUDVGWGUUv1-ccUMAZHJLej2C7aAFWY/view) **(P58~62)**

**A Bible written by RL's father Richard Sutton. It's a great masterpiece without a doubt, but it can be a bit difficult for people who start studying for the first time.**

**Markov decision process(MDP).**

**Reinforcement learning problems are mathematically described using a framework called Markov decision processes (MDPs). MDPs are the extended version of** [**Markov Chain**](https://en.wikipedia.org/wiki/Markov_chain) **which adds decisions and rewards elements to it. The word Markov here refers to that** [**Markovian property**](https://en.wikipedia.org/wiki/Markov_property) **which means that the future state is independent of any previous states history given the current state and action. This means that current state encapsulates all that is needed to decide the future state when an input action is received. This is a reasonable assumption in many problems and it simplifies things a lot. For example, in chase game, the chess board configuration after a move is being made can be decided based on the current board configuration and the action being made now and we don’t need to worry about previous chess board configurations or past actions.**

**MDP is an approach in achieving reinforcement learning to take decisions in a matrix. A grid would consist of states in the form of grids. The MDP tries to capture a world in the form of a grid by dividing it into states, actions, transition matrix, and rewards. The solution to an MDP is called a policy and the objective is to find the optimal policy for a task that MDP is imposed. Thus, any reinforcement learning task composed of a set of states, actions, and rewards that follows the Markov property would be considered an MDP.**

**In this tutorial, we will dig deep into MDPs, states, actions, rewards, and policies.**

**What is a State?**

**A State is a set of tokens that represent every condition that the agent can be in.**

**What is a Model?**

**A Model (sometimes called Transition Model) gives an action’s effect in a state. In particular, T(S, a, S’) defines a transition T where being in state S and taking an action ‘a’ takes us to state S’ (S and S’ may be same). For stochastic actions (noisy, non-deterministic) we also define a probability P(S’|S,a) which represents the probability of reaching a state S’ if action ‘a’ is taken in state S.**

**What are Actions?**

**An Action ‘a’ is set of all possible decisions. a(s) defines the set of actions that can be taken being in state S.**

**What is a Reward?**

**A Reward is a real-valued response to an action. R(s) indicates the reward for simply being in the state S. R(S,a) indicates the reward for being in a state S and taking an action ‘a’. R(S, a, S’) indicates the reward for being in a state S, taking an action ‘a’ and ending up in a state S’.**

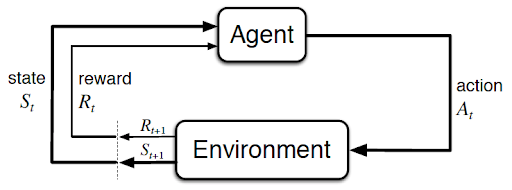
**What is a Policy?**

**A policy is a solution to the Markov Decision Process. A policy is a set of actions that are taken by the agent to reach a goal. It indicates the action ‘a’ to be taken while in state S. A policy is denoted as 'Pi'   π(s) --> ∞**

**π\* is called the optimal policy, which maximizes the expected reward. Among all the policies taken, the optimal policy is the one that optimizes to maximize the amount of reward received or expected to receive over a lifetime. For an MDP, there’s no end of the lifetime and you have to decide the end time.**

**Thus, the policy is nothing but a guide telling which action to take for a given state. It is not a plan but uncovers the underlying plan of the environment by returning the actions to take for each state.**

**Markov Decision Process (MDP) is a tuple(S,A,T,r,𝛾):**

****

***\*Rewards specify what the agent needs to achieve, not how to achieve it. (Source: Sutton and Barto,2017)***

* **‘S’ Set of observations. The agent observes the environment state as one item of this set.**
* **‘A’ Set of actions. The set of actions the agent can choose one from to interact with the environment.**
* **‘T’  - P(s' | s, a) transition probability matrix. This models what next state s' will be after the agent makes the action a while being in the current state 's'.**

**https://s3.amazonaws.com/thinkific/file_uploads/104829/images/2c4/fa4/81f/1536299488129.jpg**

* **‘r’  - P(r | s, a) reward model that models what reward the agent will receive when it performs action a when it is in state 's'.**

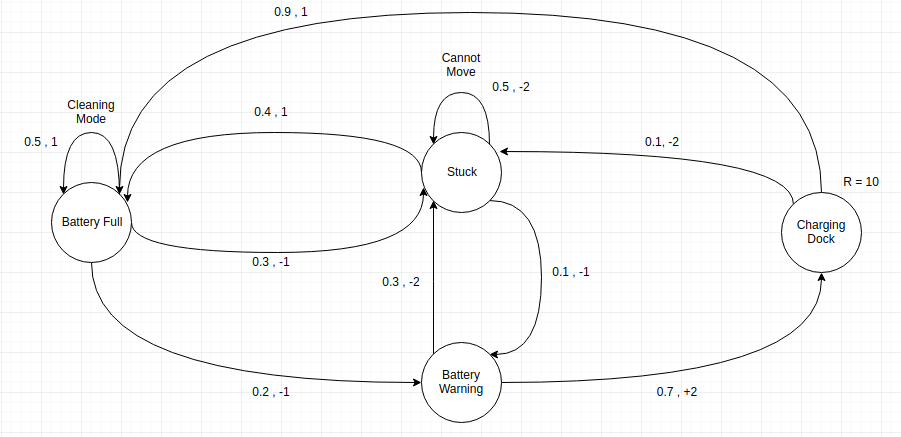
**https://s3.amazonaws.com/thinkific/file_uploads/104829/images/eb5/dc1/b82/1536299488158.jpg**

* **‘𝛾: discount factor. This factor is a numerical value between 0 and 1 that represents the relative importance between immediate and future rewards. I.e, If the agent has to select between two actions one of them will give it a high immediate reward immediately after performing the action but will lead into going to state from which the agents expect to get less future rewards than another state that can be reached after doing an action with less immediate reward?**

**https://s3.amazonaws.com/thinkific/file_uploads/104829/images/576/49f/000/1536299488188.jpg**

**Let's dive into a real-world scenario:**

**Robotic Vaccum cleaner famously known as Roomba is a machine that cleans the floor. Roomba needs to clean, avoid obstacles and find the charging station. These 4 states describe the possible positions of the robot and the action describes the direction of motion. The robot can move to the left or to the right. The first (Battery Full) and the final (Charging) states are the terminal states. The goal is to find an optimal policy that maximizes the return from any initial states.**

****

**Fig 1.0**

****

**Fig 2.0**

**Roomba lives in the closed grid. The above example is a 3\*4 grid. The grid has a START state(grid no 1,1). The purpose of the Roomba is to clean around the grid to finally reach the charging dock (grid no 4,3). Under all circumstances, the Roomba should avoid the Stuck grid ( grid no 4,2). Also the grid no (2,2 is a blocked grid, it acts like a wall hence the Roomba cross or enter it.**

**The Roomba can take any one of these actions: UP, DOWN, LEFT, RIGHT**

**Chair block the Roomba's path, i.e., if there is an obstacle in the direction the Roomba would have taken, the Roomba stays in the same place. So for example, if the Roomba says LEFT in the START grid it would stay put in the START grid and try another move.**

**First Aim: To find the longest sequence getting from START to the Charging Dock so the Romba covers maximum states. Two such sequences can be found:**

* **RIGHT RIGHT UP UP RIGHT**
* **UP UP RIGHT RIGHT RIGHT**

**Let us take the second one (UP UP RIGHT RIGHT RIGHT) for the subsequent discussion.**

**The move is now noisy. 80% of the time the intended action works correctly. 20% of the time the action Roomba takes causes it to move at right angles. For example, if the Roomba says UP the probability of going UP is 0.8 whereas the probability of going LEFT is 0.1 and probability of going RIGHT is 0.1 (since LEFT and RIGHT is right angles to UP).**

**The Roomba receives rewards each time step:-**

* **Small reward each step (can be negative then can also be termed as punishment, in the above example entering the Stuck can have a reward of -1).**
* **Big rewards come at the end (good or bad).**
* **The goal is to Maximize the sum of reward.**