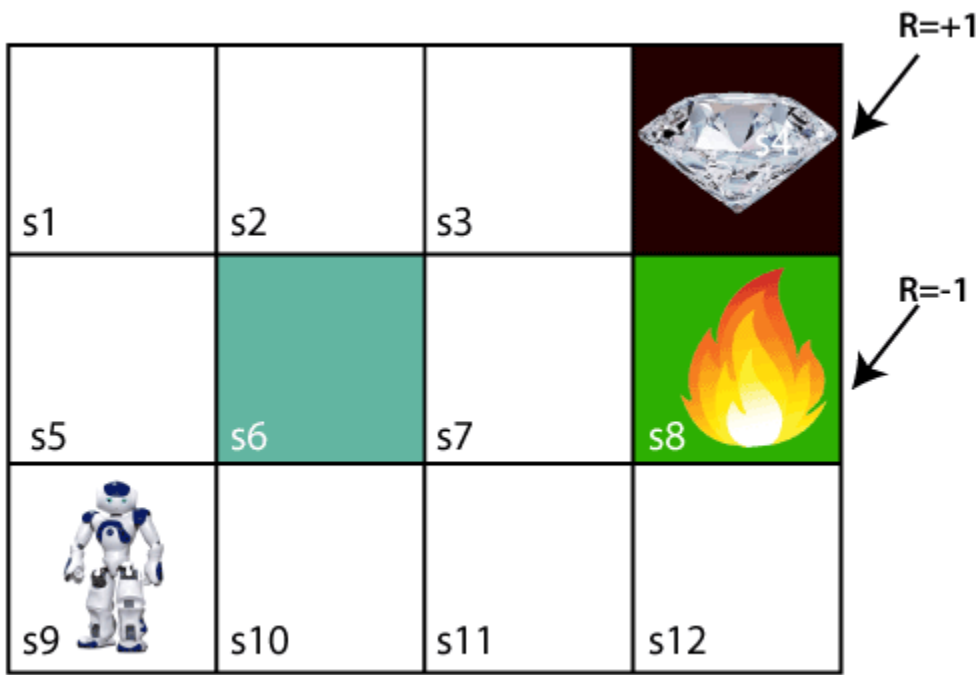


Bellman Equation

To understand the working process of the RL, we need to consider two main things:

- **Environment:** It can be anything such as a room, maze, football ground, etc.
- **Agent:** An intelligent agent such as AI robot.

Let's take an example of a maze environment that the agent needs to explore. Consider the below image:






Lorem ipsum

In the above image, the agent is at the very first block of the maze. The maze is consisting of an S_6 block, which is a **wall**, S_8 a **fire pit**, and S_4 a **diamond block**.





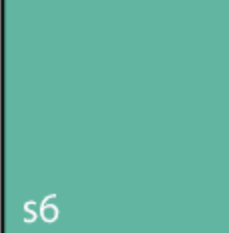

The agent cannot cross the S_6 block, as it is a solid wall. If the agent reaches the S_4 block, then get the **+1 reward**; if it reaches the fire pit, then gets **-1 reward point**. It can take four actions: **move up, move down, move left, and move right**.

The agent can take any path to reach to the final point, but he needs to make it in possible fewer steps. Suppose the agent considers the path **S9-S5-S1-S2-S3**, so he will get the +1-reward point.

The agent will try to remember the preceding steps that it has taken to reach the final step. To memorize the steps, it assigns 1 value to each previous step. Consider the below step:

<div>V=1</div> <div>s1</div>	<div>V=1</div> <div>s2</div>	<div>V=1</div> <div>s3</div>	<div>  </div> <div>s4</div>
<div>V=1</div> <div>s5</div>	<div>s6</div>	<div>s7</div>	<div>  </div> <div>s8</div>
<div>  <div>V=1</div> </div> <div>s9</div>	<div>s10</div>	<div>s11</div>	<div>s12</div>

Now, the agent has successfully stored the previous steps assigning the 1 value to each previous block. But what will the agent do if he starts moving from the block, which has 1 value block on both sides? Consider the below diagram:

s1 	s2  $V=1$	s3 $V=1$	s4 
s5 $V=1$ 	s6 	s7	s8 
s9 $V=1$	s10	s11	s12

It will be a difficult condition for the agent whether he should go up or down as each block has the same value. So, the above approach is not suitable for the agent to reach the destination. Hence to solve the problem, we will use the **Bellman equation**, which is the main concept behind reinforcement learning.

The Bellman Equation

The Bellman equation was introduced by the Mathematician **Richard Ernest Bellman in the year 1953**, and hence it is called as a Bellman equation. It is associated with dynamic programming and used to calculate the values of a decision problem at a certain point by including the values of previous states.

It is a way of calculating the value functions in dynamic programming or environment that leads to modern reinforcement learning.

The key-elements used in Bellman equations are:

- Action performed by the agent is referred to as "a"
- State occurred by performing the action is "s."
- The reward/feedback obtained for each good and bad action is "R."
- A discount factor is Gamma " γ ."

The Bellman equation can be written as:

1. $V(s) = \max [R(s,a) + \gamma V(s')]$

Where,

$V(s)$ = value calculated at a particular point.

$R(s,a)$ = Reward at a particular state s by performing an action.

γ = Discount factor

$V(s')$ = The value at the previous state.

In the above equation, we are taking the max of the complete values because the agent tries to find the optimal solution always.

So now, using the Bellman equation, we will find value at each state of the given environment. We will start from the block, which is next to the target block.

For 1st block:

$V(s_3) = \max [R(s,a) + \gamma V(s')]$, here $V(s') = 0$ because there is no further state to move.

$V(s_3) = \max[R(s,a)] \Rightarrow V(s_3) = \max[1] \Rightarrow \mathbf{V(s_3) = 1.}$

For 2nd block:

$V(s_2) = \max [R(s,a) + \gamma V(s')]$, here $\gamma = 0.9$ (lets), $V(s') = 1$, and $R(s, a) = 0$, because there is no reward at this state.

$V(s_2) = \max[0.9(1)] \Rightarrow V(s) = \max[0.9] \Rightarrow \mathbf{V(s_2) = 0.9}$

For 3rd block:

$V(s_1) = \max [R(s,a) + \gamma V(s')]$, here $\gamma = 0.9$ (lets), $V(s') = 0.9$, and $R(s, a) = 0$, because there is no reward at this state also.

$V(s_1) = \max[0.9(0.9)] \Rightarrow V(s_3) = \max[0.81] \Rightarrow \mathbf{V(s_1) = 0.81}$

For 4th block:

$V(s_5) = \max [R(s,a) + \gamma V(s')]$, here $\gamma = 0.9$ (lets), $V(s') = 0.81$, and $R(s, a) = 0$, because there is no reward at this state also.


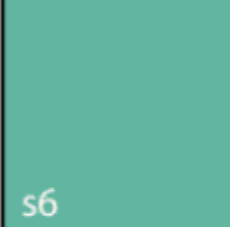


$V(s_5) = \max[0.9(0.81)] \Rightarrow V(s_5) = \max[0.73] \Rightarrow \mathbf{V(s_5) = 0.73}$

For 5th block:


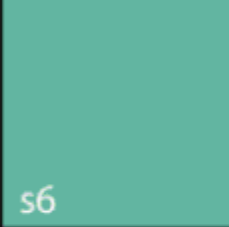


$V(s_9) = \max [R(s,a) + \gamma V(s')]$, here $\gamma = 0.9$ (lets), $V(s') = 0.73$, and $R(s, a) = 0$, because there is no reward at this state also.

$$V(s_9) = \max[0.9(0.73)] \Rightarrow V(s_4) = \max[0.81] \Rightarrow \mathbf{V(s_4) = 0.66}$$


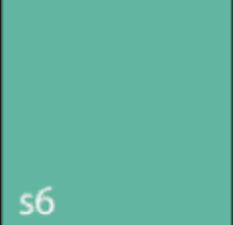

Consider the below image:

$V=0.81$ s1	$V=0.9$ s2	$V=1$ s3	 s4
$V=0.73$ s5	 s6	s7	 s8
 $V=0.66$ s9	s10	s11	s12

Now, we will move further to the 6th block, and here agent may change the route because it always tries to find the optimal path. So now, let's consider from the block next to the fire pit.

$V=0.81$ s1	$V=0.9$ s2	$V=1$ s3	 s4
$V=0.73$ s5	 s6	 s7	 s8
$V=0.66$ s9	s10	s11	s12

Now, the agent has three options to move; if he moves to the blue box, then he will feel a bump if he moves to the fire pit, then he will get the -1 reward. But here we are taking only positive rewards, so for this, he will move to upwards only. The complete block values will be calculated using this formula. Consider the below image:

$V=0.81$ s1	$V=0.9$ s2	$V=1$ s3	 s4
$V=0.73$ s5	 s6	$V=0.9$ s7	 s8
$V=0.66$ s9	$V=0.73$ s10	$V=0.81$ s11	$V=0.73$ s12