IS 7033: Artificial Intelligence and Machine Learning

Dr. Peyman Najafirad (Paul Rad)
Associate Professor
Cyber Analytics and AI
210.872.7259

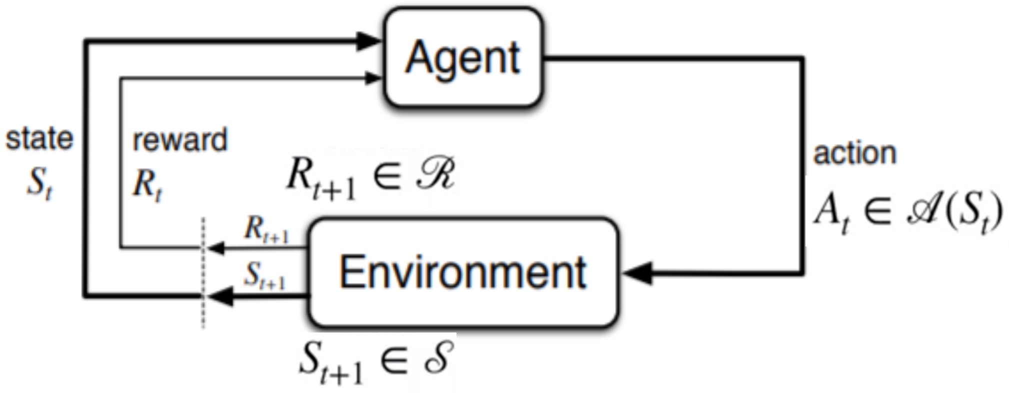
https://github.com/paulNrad/ProbabilisticGraphModels

Reinforcement Learning

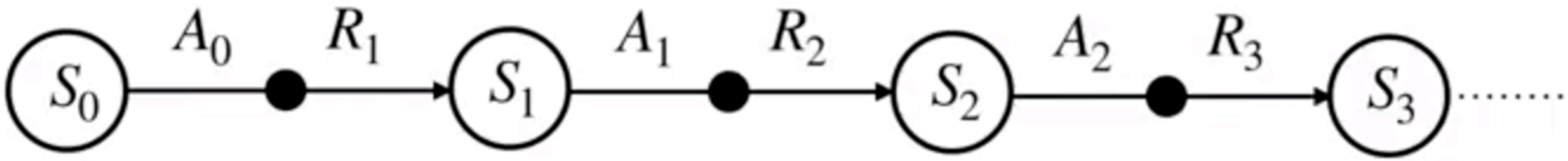
Read Reinforcement Learning by Richard Sutton Chapter 3 and 4

Reinforcement Learning Summary

- The reinforcement learning (RL) framework is characterized by an **agent** learning to interact with its **environment**.
- At each time step, the agent receives the environment's **state** (*the environment present a situation to the agent)*, and the agent must choose an appropriate **action** in response. One time step later, the agent receives a **reward** (*the environment indicates whether the agent has responded appropriately to the state*) and a new **state**.
- All agents have the goal to maximize expected **cumulative reward**, or the expected sum of rewards attained over all time steps.



The agent-environment interaction in reinforcement learning. (Source: Sutton and Barto, 2017)



Episodic vs. Continuing Tasks

- **Continuing tasks** are tasks that continue forever, without end.

- **Episodic tasks** are tasks with a well-defined starting and ending point. In this case, we refer to a complete sequence of interaction, from start to finish, as an **episode**. Episodic tasks come to an end whenever the agent reaches a **terminal state**.

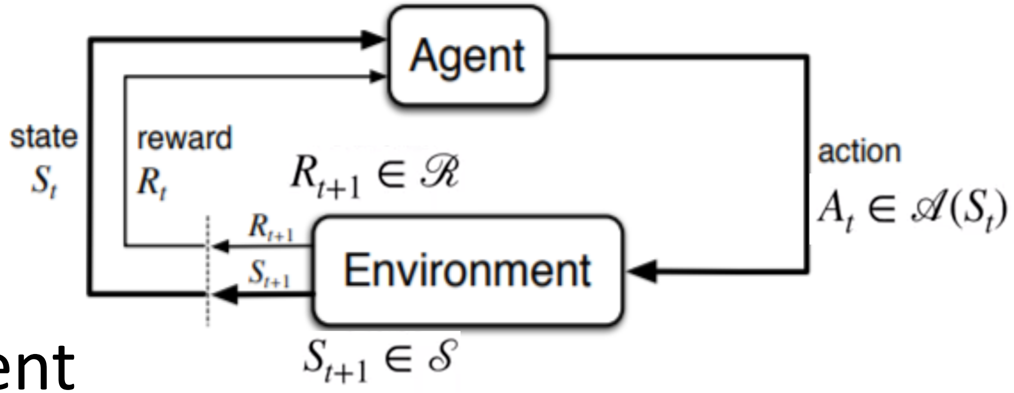
Goals, Cumulative Reward, and Discounted Return

- The **return at time step** *t* is $G_t := R_{t+1} + R_{t+2} + R_{t+3} + \ldots$
- The agent selects actions with the goal of maximizing expected (discounted) return.
- The **discounted return at time step** *t* is $G_t := R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots$
  - The **discount rate** γ is something that you set, to refine the goal that you have the agent. It must satisfy $0 \leq \gamma \leq 1$.
  - If $\gamma=0$, the agent only cares about the most immediate reward.
  - If $\gamma=1$, the return is not discounted.
  - For larger values of γ, the agent cares more about the distant future. Smaller values of γ result in more extreme discounting, where - in the most extreme case - agent only cares about the most immediate reward.

MDPs and One-Step Dynamics

A (finite) Markov Decision Process (MDP) is defined by:

- A (finite) set of states S
- A (finite) set of actions A
- A set of rewards R
- The one-step dynamics of the environment



p(s', r | s, α) = P(S_{t+1} = s', R_{t+1} = r | S_t = s, A_t = α) for all s, s', α, and r

- A discount rate γ $\mathcal{E}$ $[0,1]$

We can start to think of the solution as a series of actions that need to be learned by the agent towards the pursuit of a goal.

Policies

A policy determines how an **agent chooses an action in response to the current state**. In other words, it specifies how the agent responds to situations that the environment has presented.

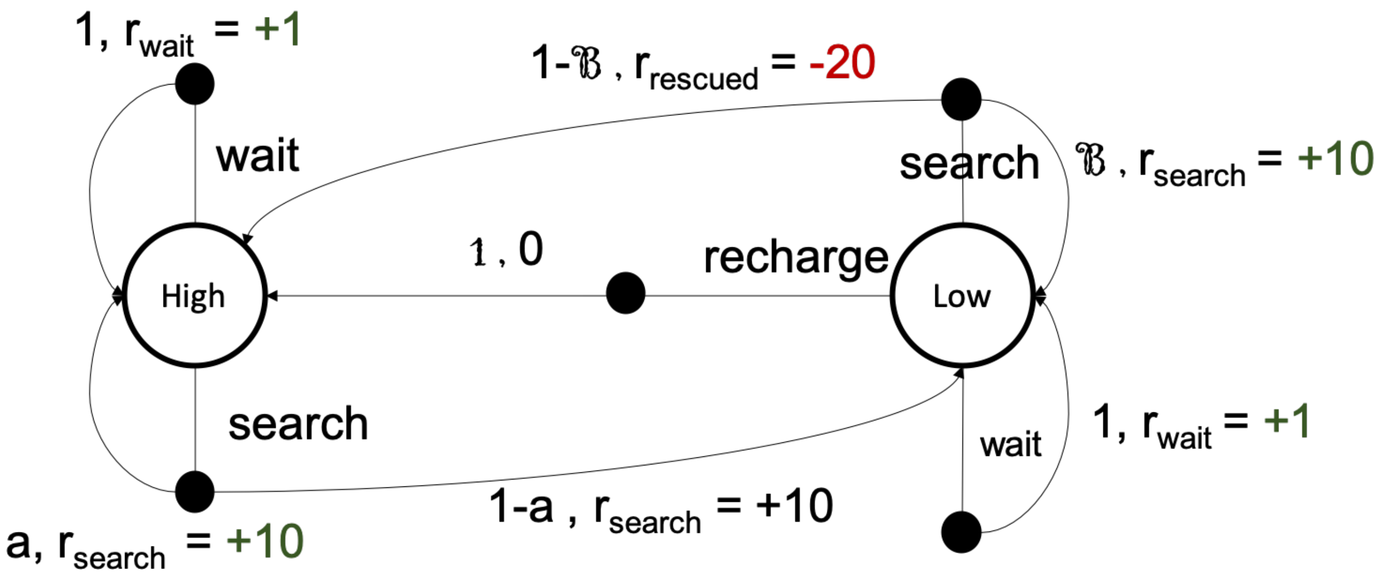
- A **deterministic policy** is mapping from the set of environment states to the set of possible actions:

$$\pi : S \rightarrow A$$

- A **stochastic policy** is a mapping $\pi : S \text{ x } A \rightarrow [0,1]$

$$\pi\ (\alpha \mid s) = P\ (A_t = \alpha \mid S_t = s)$$

Policies



Cleaning Robot

Stochastic Policy

π (recharge | low) = 0.5
π (search | low) = 0.1
π (wait | low) = 0.4
π (search | high) = 0.9
π (wait | high) = 0.1

Deterministic Policy

π (recharge | low) = 1
π (search | high) = 1

Now that we know how to establish a policy, what step can we take to make sure the agent's policy is the best one (optimal policy).

Question: Consider a different stochastic policy where:

π (recharge | low) = 0.3

π (search | low) = 0.2

π (wait | low) = 0.5

π (search | high) = 0.6

π (wait | high) = 0.4

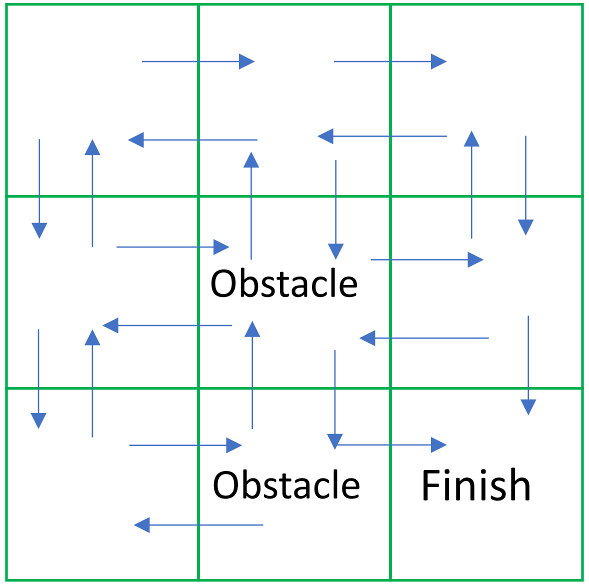
Which of the following statements are true, if the agent follows the policy?

a) If the battery is low, the agent will always decide to wait for cans.
b) If the battery level is high, the agent chooses to search for a can with 60% probability, and otherwise waits for a can.
c) If the battery level is low, the agent is most likely to decide to wait for cans.

Grid world

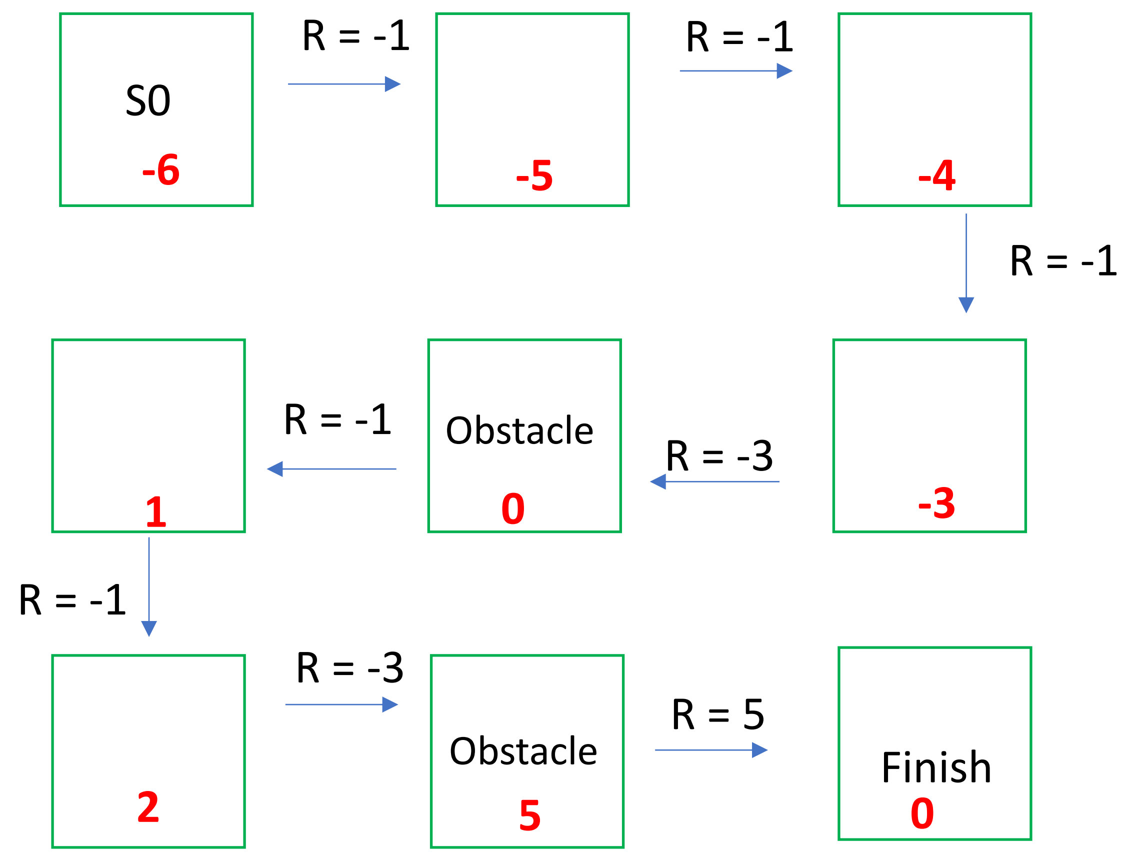
Let’s consider:

- The agent can only move up, down, left or right, and can only take actions that lead in to not fall off the grid.
- The goal of the agent to get to the bottom right hand corner of the world as quickly as possible.
- The agent receives a rewards of negative one for most transitions. If it leads to a mountain rewards of 3 and the finish state rewards of 5.



MDP and Agent Policy

Let's choose a policy that the agent visits every state in a roundabout manner.



Cumulative Return for S0 ➔ (-1) + (-1) + (-1) + (-3) + (-1) + (-1) + (-3) + (5) = -6

State-Value Function

For each state, the **state-value function** yields the **expected return**, if the agent started in that state, and then followed the policy for all times steps.

-6	**-5**	**-4**
1	**0**	**-3**
2	**5**	**0**

We call $\boldsymbol{v_\pi}$ the state-value function for policy π

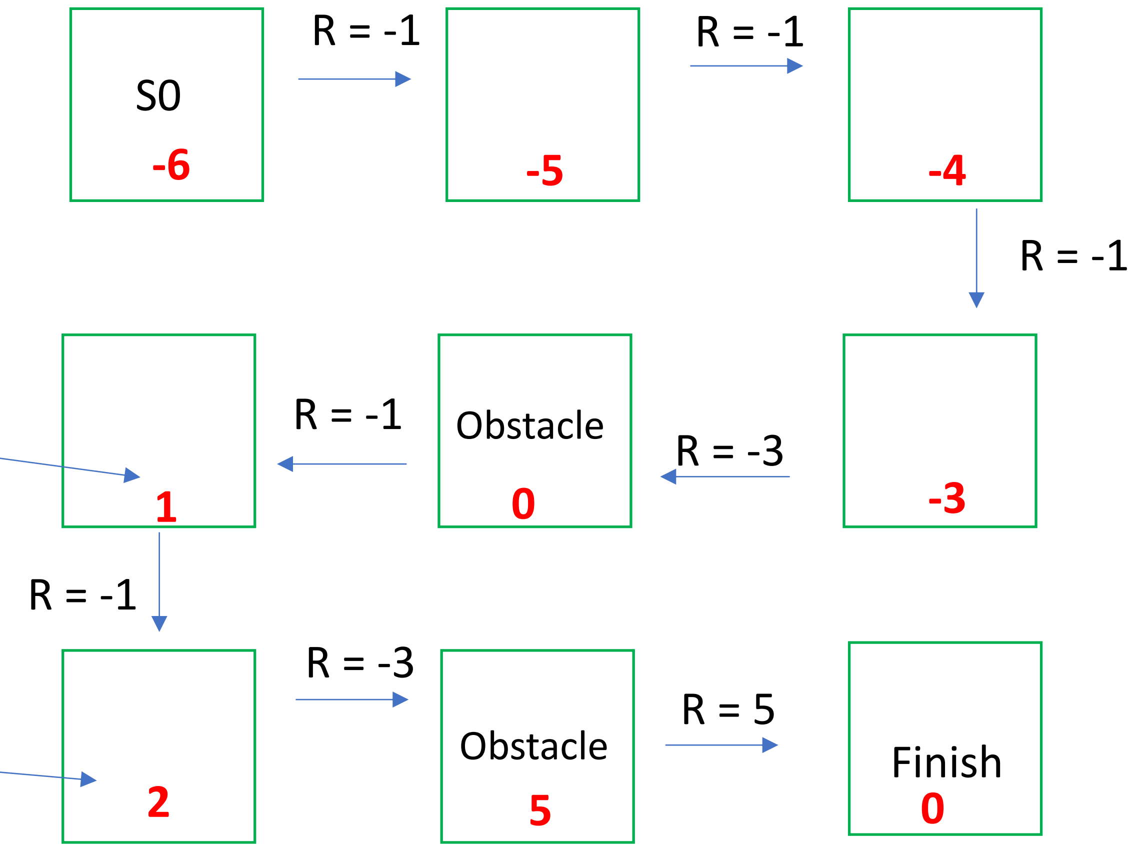
The value of state s under a policy π is:

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

For each state s

It yields the **expected return** If the agent starts in **state s** and then uses **the policy** to choose its actions for all time steps.

Bellman Equations



The value of **<u>any state</u>** **<u>(1)</u>**

=

the immediate reward (-1)

+

The value of of the state that follows (2)

Discount rate is 1

Bellman Expected Equation

$$v_\pi (s) = E_\pi [R_{t+1} + \gamma \, v_\pi (S_{t+1}) \mid S_t = s]$$

The value of any state s = the expected value of the immediate rewards
+
The discounted value of the state that follows under that policy.

Question: **State-Value Functions for $\boldsymbol{\pi}'$**

You will calculate the value function corresponding to a particular policy

<u>Deterministic policy</u>

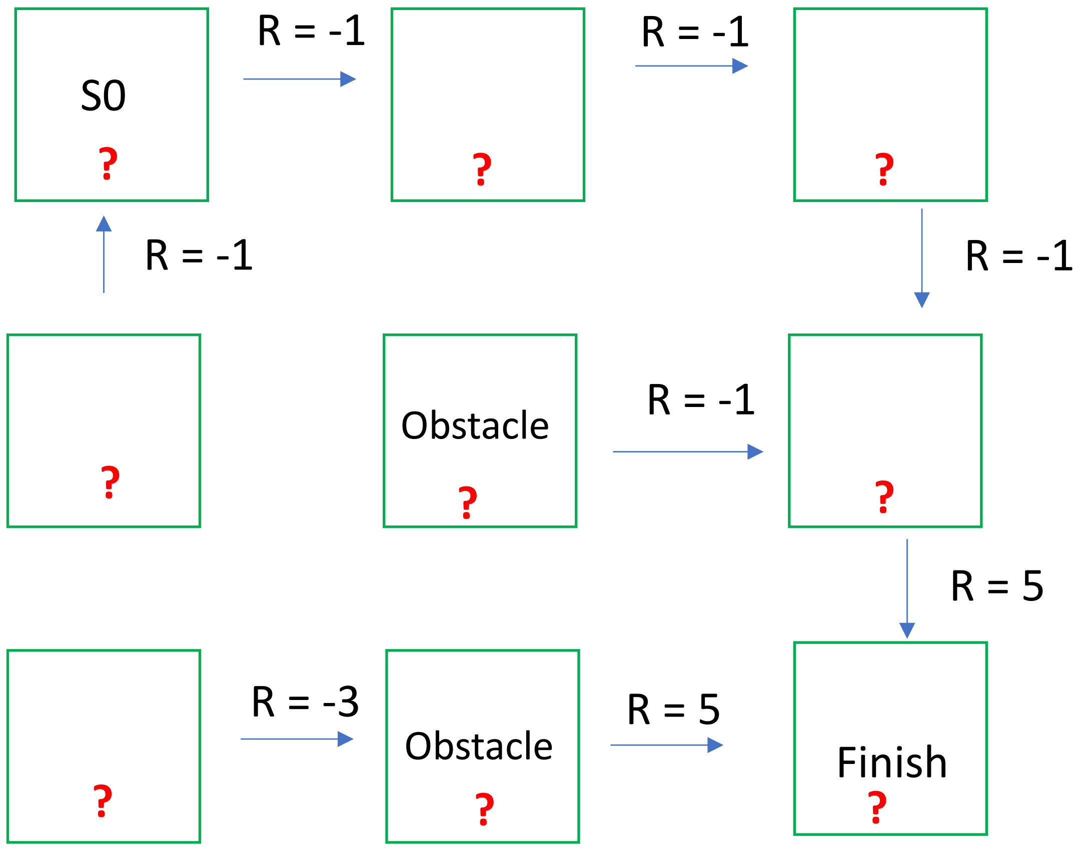
$\pi(s_1)$=right
$\pi(s_2)$=right
$\pi(s_3)$=down
$\pi(s_4)$=up
$\pi(s_5)$=right
$\pi(s_6)$=down
$\pi(s_7)$=right
$\pi(s_8)$=right

Assume γ=1

<u>Questions:</u>

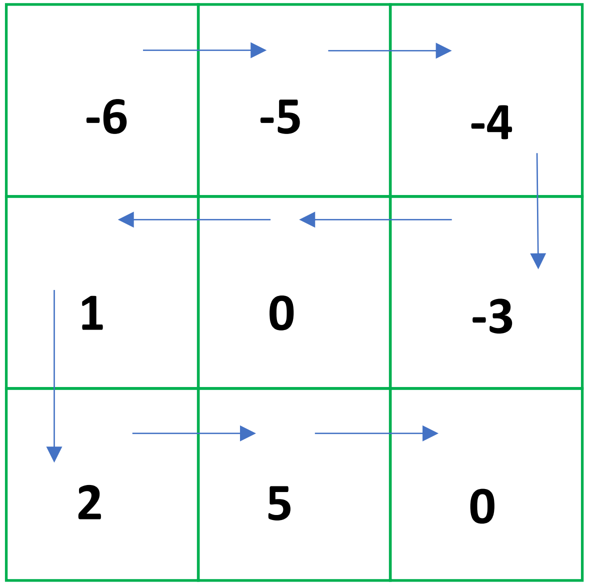
What is $v_\pi(s_4)$?

What is $v_\pi(s_1)$?

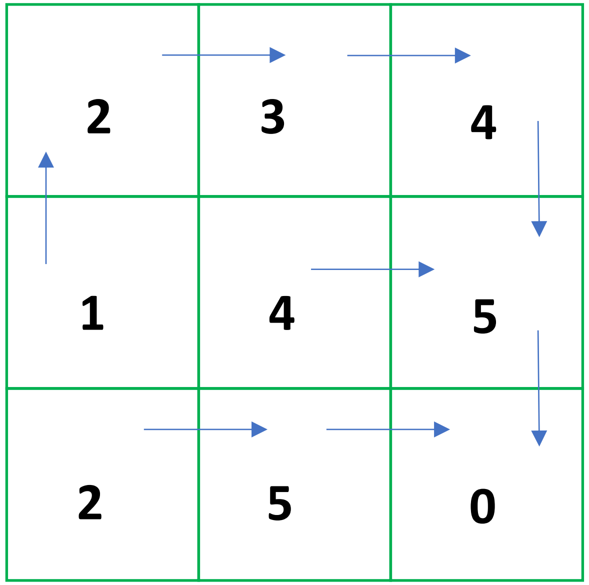


State value functions for different policies

State value function for policy π



State value function for policy π'



$$\pi' \geq \pi \; if \; and \; only \; if \; v_{\pi'}(s) \geq v_{\pi}(s) \text{ for all s } \in S$$

Optimal policy

Definition

$\boldsymbol{\pi}' \geq \boldsymbol{\pi}$ ***if and only if*** $\boldsymbol{v}_{\pi'}$**(s)** $\geq \boldsymbol{v}_{\pi}$**(s) for all s** $\in \boldsymbol{S}$

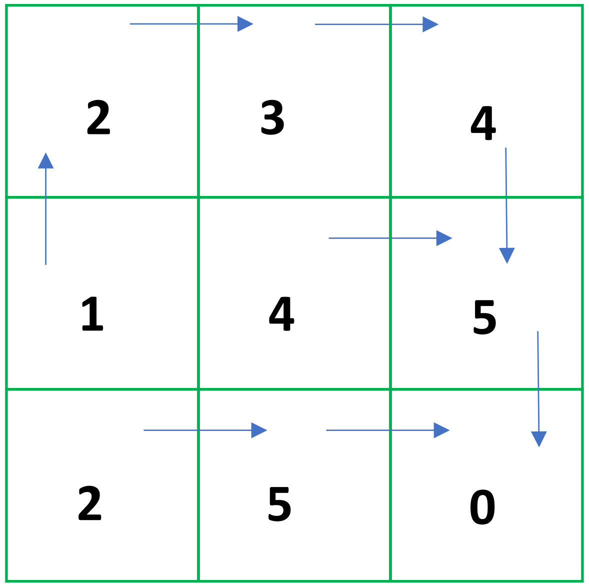
Definition

An optimal policy $\boldsymbol{\pi}^*$ satisfies $\boldsymbol{\pi}^* \geq \boldsymbol{\pi}$ for all $\boldsymbol{\pi}$

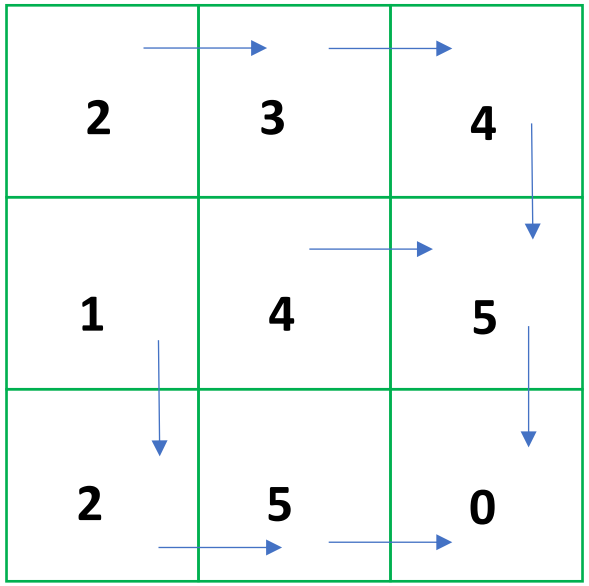
The optimal state value function is denoted v^*

Both are Optimal Policies

State value function for policy π'



State value function for policy π''



Action Value Function $q_\pi(s, a)$

We call $\boldsymbol{v_\pi}$ the state-value function for policy π

The value of state s under a policy π is:

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

For each state s

It yields the **expected return** If the agent starts in **state s** and then uses **the policy** to choose its actions for all future time steps.

We call $q_{\boldsymbol{\pi}}$ the action-value function for policy π

The value of taking action a under a policy π is:

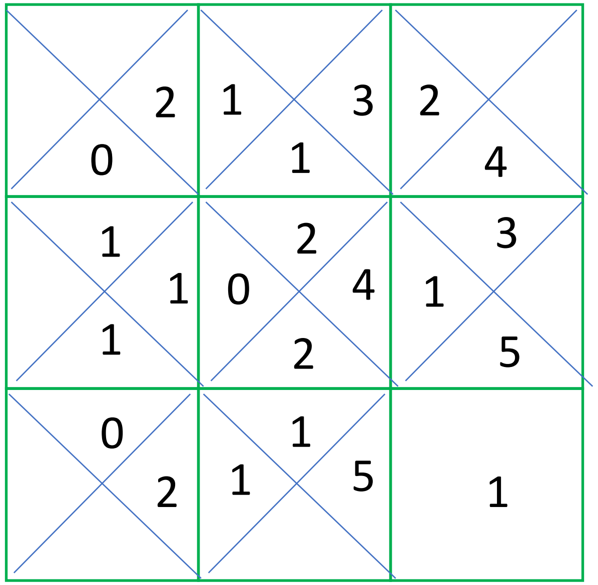
$$q_\pi(s, \alpha) = E_\pi[G_t | S_t = s, A_t = \alpha]$$

For each state s and action a

It yields the **expected return** If the agent starts in **state s** and then chooses **action a** and then uses **the policy** to choose its actions for all future time steps.

Action Value for Policy π

Action value function for policy π'



Optimal Policy

- The agent interacts with the environment. From that interactions, it **estimates** the **optimal action value function**. Then the agent uses that **value function** to get the **optimal policy**.

Interaction ➔ q_* ➔ π_*

Action value function for policy π'