HARVARD
School of Engineering
and Applied Sciences

# Reinforcement Learning

## Monte Carlo Methods

Pablo Ruiz Ruiz

Deep Learning Intern Researcher @ Harvard University

# Index

- Monte Carlo Prediction
  - First-visit / Every-visit
  - Greedy / $\varepsilon$-greedy
  - GLIE
- Monte Carlo Control
  - Incremental Update
  - Constant step size

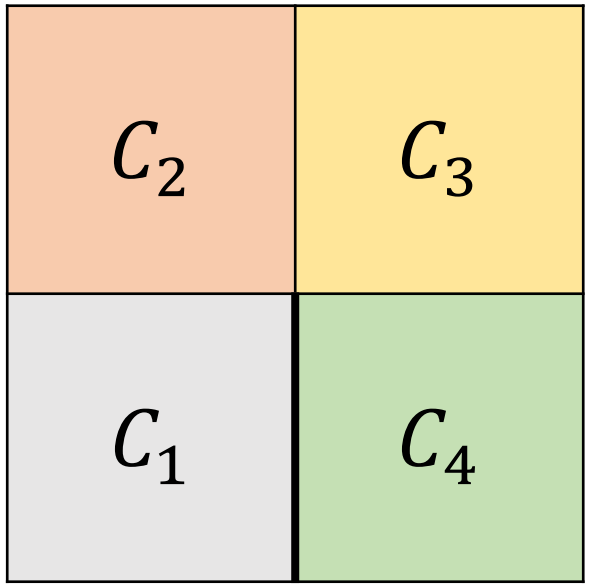
# Monte Carlo Methods

## Monte Carlo Prediction

**How good is my current policy?**

# Environment Example

**States**



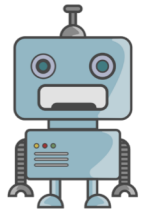

**Actions**

- $a_1$
- $a_2$
- $a_3$
- $a_4$

**Rewards**

- $+10$ for reaching $C_4$
- $-1$ otherwise

We call the states as $C_i$ because in the formulas they use $s_i$ to represent the state at each time step.
Thus, **there are 4 possible states** in this environment



**Agent**

# Environment Example

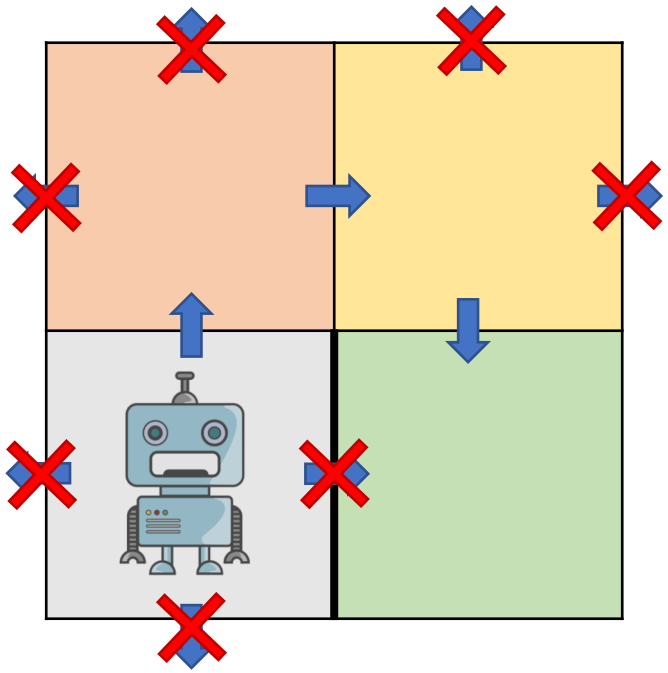


Table of values and Counter Table

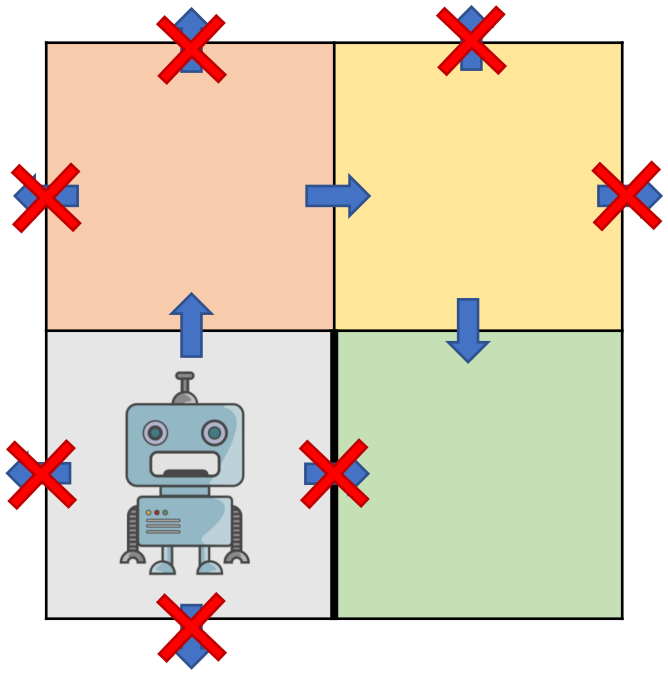
| | $a_1$ | $a_2$ | $a_3$ | $a_4$ |
|---|---|---|---|---|
| $c_1$ | | | | |
| $c_2$ | | | | |
| $c_3$ | | | | |

X represents that if the agent takes that action, it will remain in the same cell in the next state

The Table of values is trying to answer the natural question: *which action is the best one at each state?*

Counter Table is keeping track of how many time each state has been visited

# Environment Example



Optimal policy $\pi^*$: $c_1$ ↑ $c_2$ → $c_3$ ↓ $c_4$

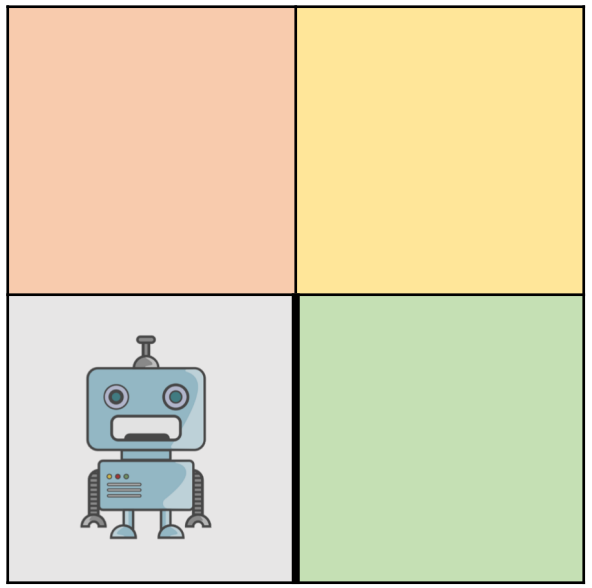
The agent has to find this policy by **interacting** with the environment

# Monte Carlo Methods

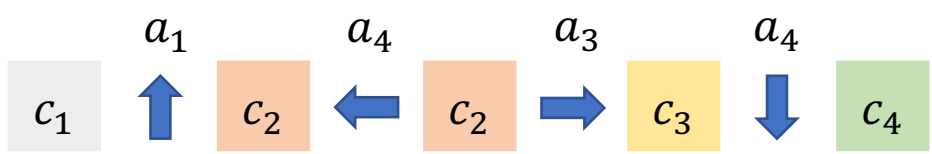
## Monte Carlo Prediction

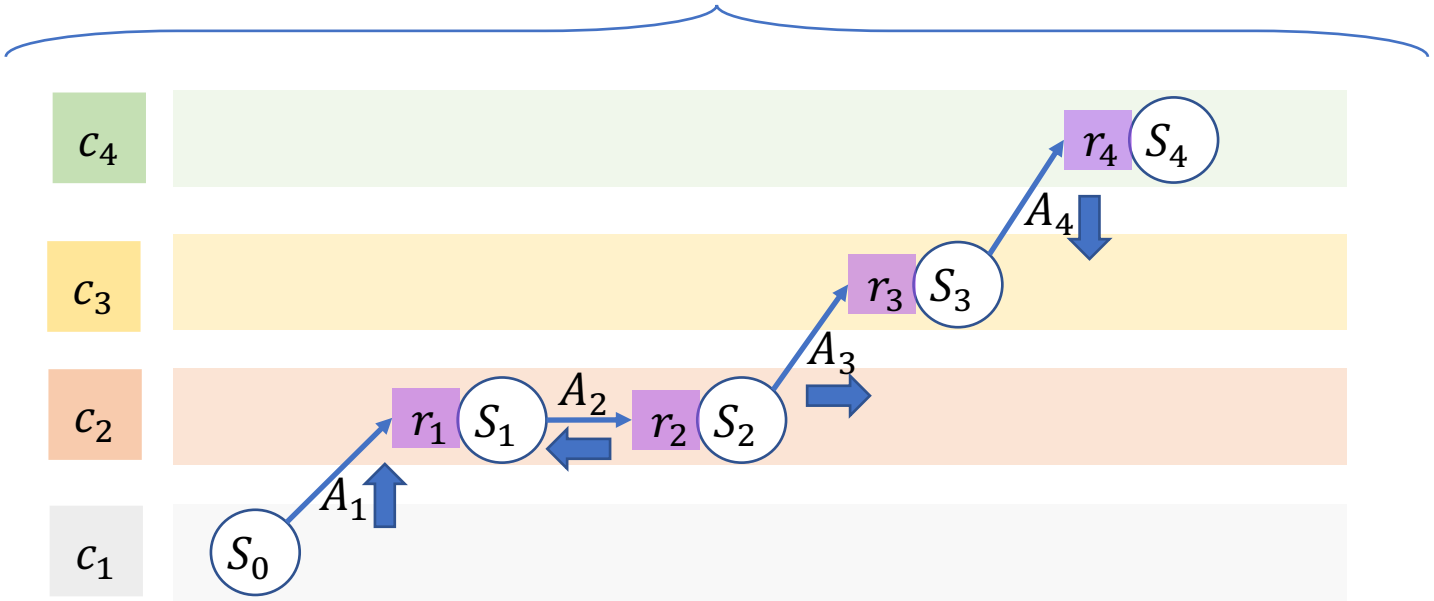
# Monte Carlo Prediction

**How do we update the table?**



Episode 1: Monte Carlo Methods run ENTIRE episodes




Rewards

| | |
|---|---|
| $r_1$ | -1 |
| $r_2$ | -1 |
| $r_3$ | -1 |
| $r_4$ | +10 |

# Monte Carlo Prediction

**How do we update the table?**

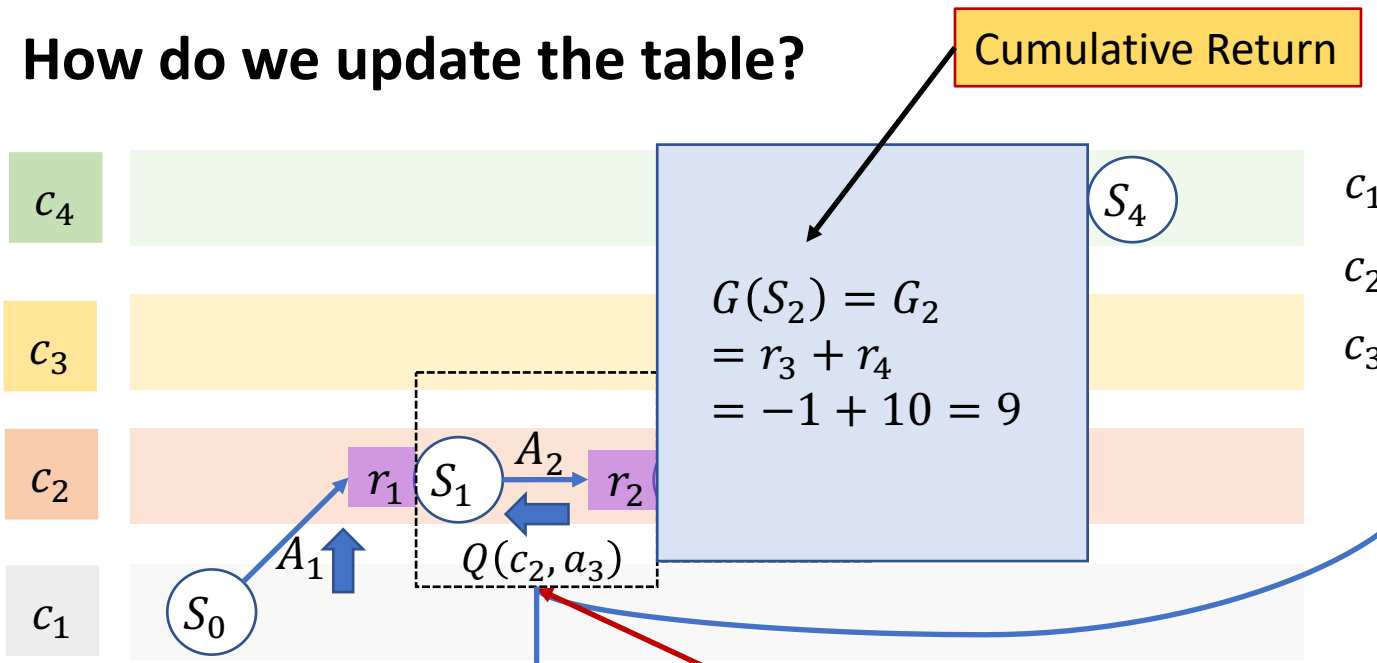



Table of values

| | $a_1$ ↑ | $a_2$ ↓ | $a_3$ ← | $a_4$ → |
|---|---|---|---|---|
| $c_1$ | | | | |
| $c_2$ | | | +8 | |
| $c_3$ | | | | |

Counter Table $N(s, a)$

| | $a_1$ ↑ | $a_2$ ↓ | $a_3$ ← | $a_4$ → |
|---|---|---|---|---|
| $c_1$ | | | | |
| $c_2$ | | | +1 | |
| $c_3$ | | | | |

$N(c_2, a_3)$

Rewards

| | |
|---|---|
| $r_1$ | -1 |
| $r_2$ | -1 |
| $r_3$ | -1 |
| $r_4$ | +10 |

**Update Rule:** $Q(c_2, a_3) = r_2 + G_2 = -1 + 9 = 8$

**General Update Rule:** $Q(s_t, a_{t+1}) = r_{t+1} + G_{t+1}$

# Monte Carlo Prediction

**How do we update the table?**

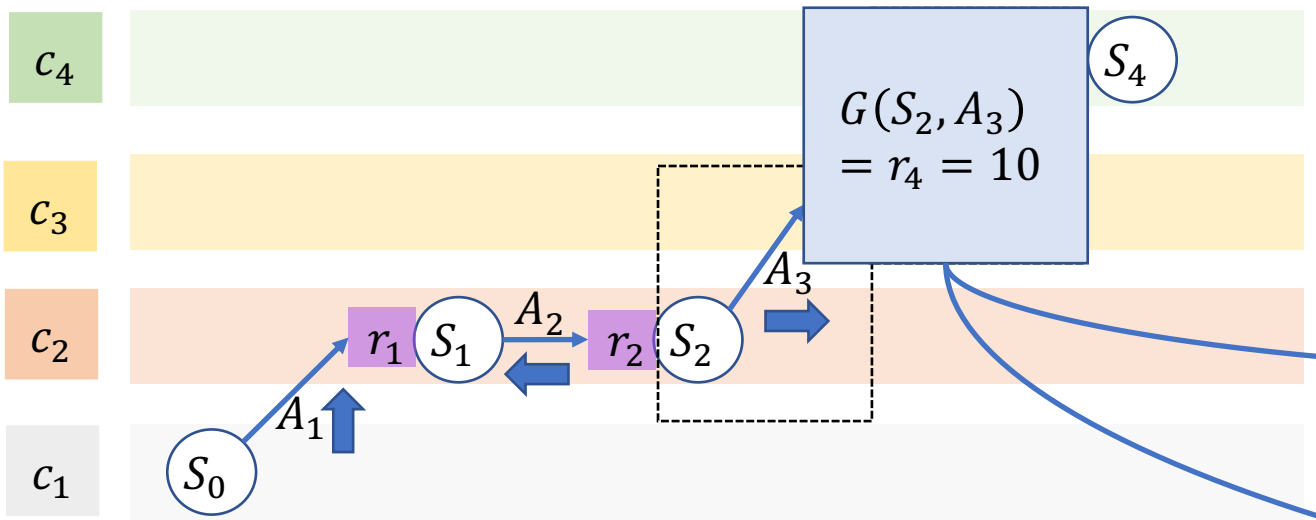



Table of values

| | $a_1$ ↑ | $a_2$ ↓ | $a_3$ ← | $a_4$ → |
|---|---|---|---|---|
| $c_1$ | | | | |
| $c_2$ | | | +8 | +9 |
| $c_3$ | | | | |

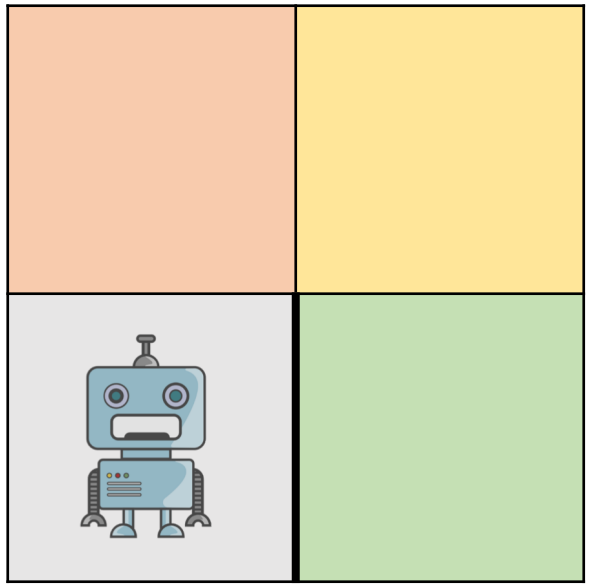
Counter Table $N(s, a)$

| | $a_1$ ↑ | $a_2$ ↓ | $a_3$ ← | $a_4$ → |
|---|---|---|---|---|
| | | | | |
| | | | +1 | +1 |
| | | | | |

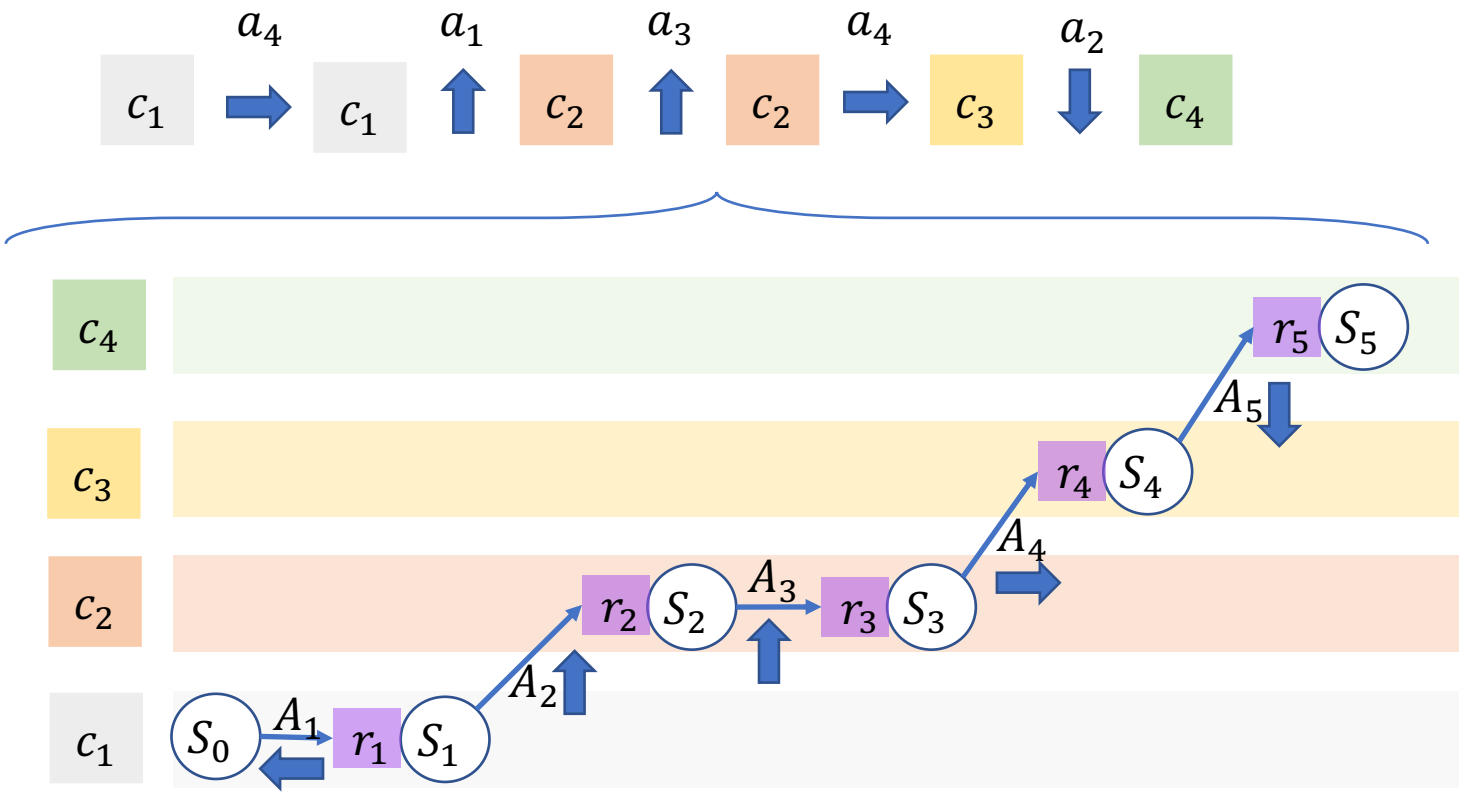
This was just 1 episode

Let's take a look at another episode

# Monte Carlo Prediction



Episode 2: Monte Carlo Methods run ENTIRE episodes




Rewards

| | |
|---|---|
| $r_1$ | -1 |
| $r_2$ | -1 |
| $r_3$ | -1 |
| $r_4$ | -1 |
| $r_5$ | +10 |

# Monte Carlo Prediction

**Table start with the values of previous episodes**

**How do we update the table?**

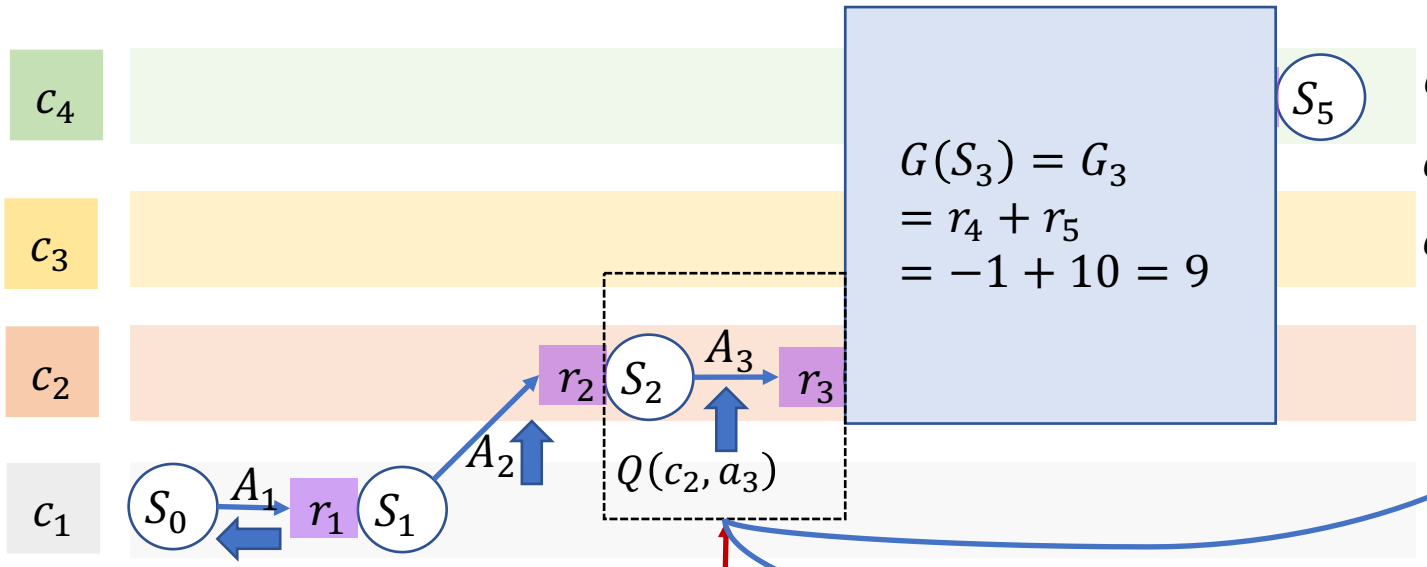



Table of values

| | $a_1$ ↑ | $a_2$ ↓ | $a_3$ ← | $a_4$ → |
|---|---|---|---|---|
| $c_1$ | | | | |
| $c_2$ | +8 | | +8 | +9 |
| $c_3$ | | | | |

Counter Table $N(s, a)$

| | $a_1$ ↑ | $a_2$ ↓ | $a_3$ ← | $a_4$ → |
|---|---|---|---|---|
| $c_1$ | | | | |
| $c_2$ | +1 | | +1 | +1 |
| $c_3$ | | | | |

**Update Rule:** $Q(c_2, a_1) = r_3 + G_3 = -1 + 9 = 8$

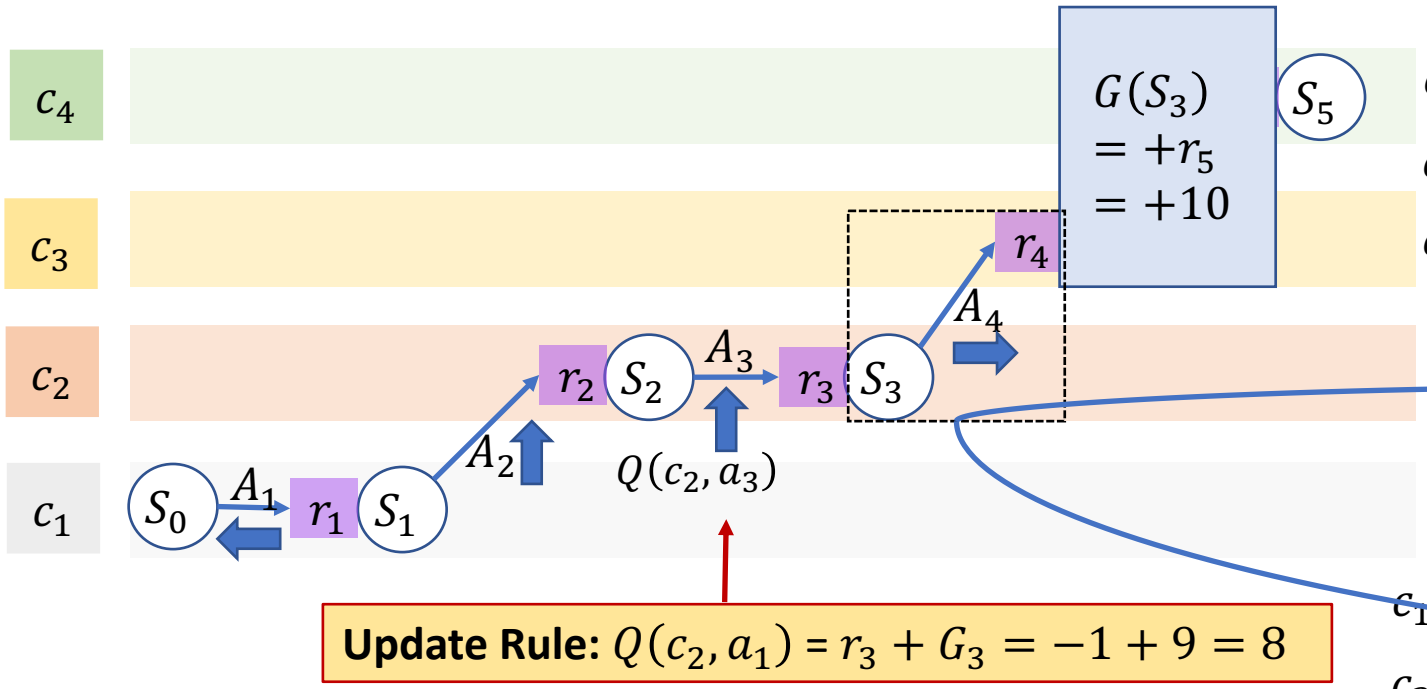
Rewards

| | |
|---|---|
| $r_1$ | -1 |
| $r_2$ | -1 |
| $r_3$ | -1 |
| $r_4$ | +10 |

# Monte Carlo Prediction

**Table start with the values of previous episodes**

**How do we update the table?**




**Update Rule:** $Q(c_2, a_1) = r_3 + G_3 = -1 + 9 = 8$

Table of values

| | $a_1$ ↑ | $a_2$ ↓ | $a_3$ ← | $a_4$ → |
|---|---|---|---|---|
| $c_1$ | | | | |
| $c_2$ | +8 | | +8 | +9 |
| $c_3$ | | | | |

Counter Table $N(s, a)$

| | $a_1$ ↑ | $a_2$ ↓ | $a_3$ ← | $a_4$ → |
|---|---|---|---|---|
| $c_1$ | | | | |
| $c_2$ | +1 | | +1 | +1 |
| $c_3$ | | | | |

**WHAT DO WE DO NOW?**

# Monte Carlo Methods

First-Visit
vs
Every-Visit

# Monte Carlo First Visit

**Table start with the values of previous episodes**

**How do we update the table?**

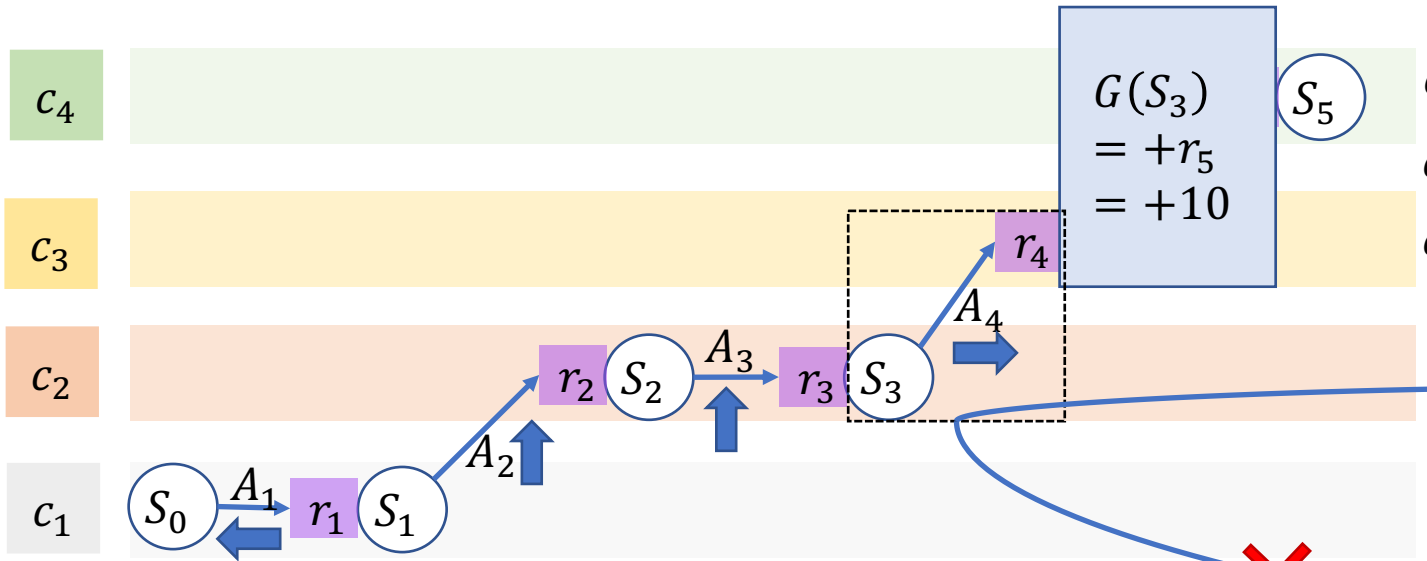



Table of values

| | $a_1$ ↑ | $a_2$ ↓ | $a_3$ ← | $a_4$ → |
|---|---|---|---|---|
| $c_1$ | | | | |
| $c_2$ | +8 | | +8 | +9 |
| $c_3$ | | | | |

Counter Table $N(s,a)$

| | $a_1$ ↑ | $a_2$ ↓ | $a_3$ ← | $a_4$ → |
|---|---|---|---|---|
| $c_1$ | | | | |
| $c_2$ | +1 | | +1 | +1 |
| $c_3$ | | | | |

**Algorithm 9:** First-Visit MC Prediction (*for action values*)

**Input:** policy $\pi$, positive integer *num_episodes*
**Output:** value function $Q$ ($\approx q_\pi$ if *num_episodes* is large enough)
Initialize $N(s,a) = 0$ for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$
Initialize $returns\_sum(s,a) = 0$ for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$
**for** $i \leftarrow 1$ ***to*** *num_episodes* **do**
    Generate an episode $S_0, A_0, R_1, \ldots, S_T$ using $\pi$
    **for** $t \leftarrow 0$ ***to*** $T-1$ **do**
        **if** $(S_t, A_t)$ *is a first visit (with return* $G_t$*)* **then**
            $N(S_t, A_t) \leftarrow N(S_t, A_t) + 1$
            $returns\_sum(S_t, A_t) \leftarrow returns\_sum(S_t, A_t) + G_t$
    **end**
**end**
$Q(s,a) \leftarrow returns\_sum(s,a)/N(s,a)$ for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$
**return** $Q$

# Monte Carlo First Visit

**Table start with the values of previous episodes**

**How do we update the table?**

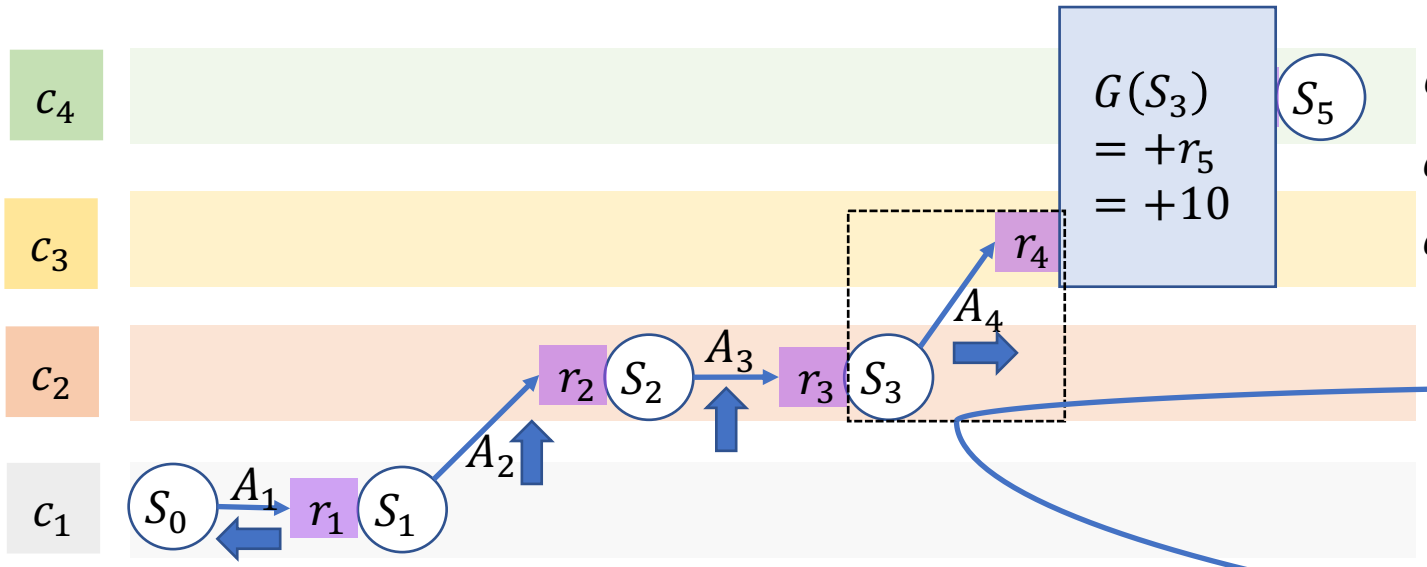



Table of values

| | $a_1$ ↑ | $a_2$ ↓ | $a_3$ ← | $a_4$ → |
|---|---|---|---|---|
| $c_1$ | | | | |
| $c_2$ | +8 | | +8 | +9+9=18 |
| $c_3$ | | | | |

Counter Table $N(s,a)$

| | $a_1$ ↑ | $a_2$ ↓ | $a_3$ ← | $a_4$ → |
|---|---|---|---|---|
| $c_1$ | | | | |
| $c_2$ | +1 | | +1 | +1+1 = 2 |
| $c_3$ | | | | |

# NOTE 1

In Reinforcement Learning, one of the fundamental concepts are the value functions.

The **action-value functions**, $q_\pi\ (s, a)$, are the ones used for the control problem. $q_\pi\ (s, a)$ is the expected return of taking a particular action in a particular state, and then follow the current policy $\pi$

That's why the value table commonly receives the name of **Q-Table**

# NOTE 2

After running all the episodes, we end up with a final picture for our Q-Table. We have been all the time **following a given policy**, **(which was acting randomly)**, that tells the agent what to do at each particular state.

So the only information we have from this process is to know **how good is the policy we are following**, how good is to act random at every state

$Q(s, a)$

| | | | |
|---|---|---|---|
| +5 | +4 | +2 | +4 |
| +7 | +5 | +8 + 7.8 | +9 |
| +10 | +7 | +8 | +8 |

# Monte Carlo Methods

## Monte Carlo Control

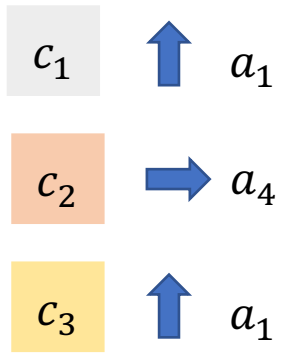
**Which is the best policy?**

# NOTE 3

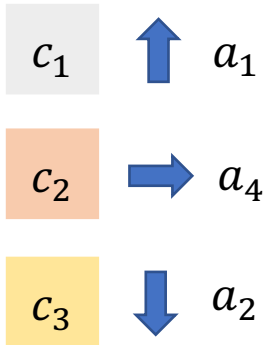
$Q(s, a)$

| | | | |
|---|---|---|---|
| +7 | +6 | +5 | +6 |
| +8 | +7 | +8 | +9 |
| +10 | +7 | +8 | +8 |

$\pi'$



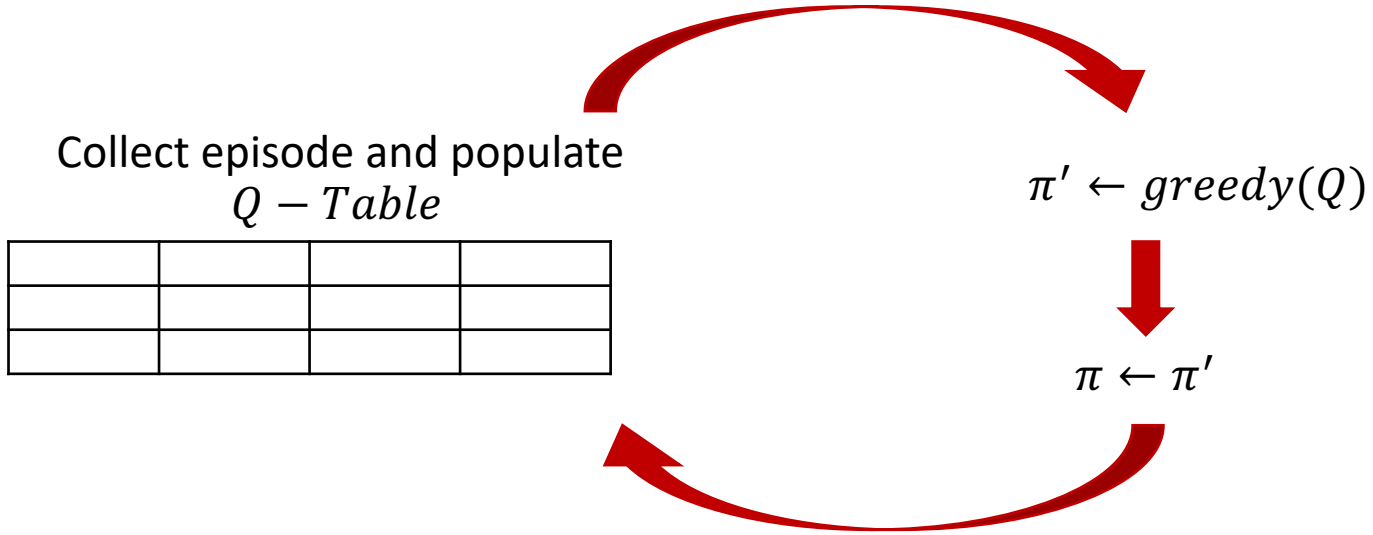

Optimal policy $\pi^*$:




The resulting Q-Table could be used to find a better policy that the one we have been following. **By taking the maximum of each row, we know which action is the best one to make at each possible state**.

What we are sure is that the new policy found is at least $\pi' \geq \pi$

We are constructing the policy that is **greedy** with respect to the Q-Table


Collect episode and populate
$Q - Table$
$\pi' \leftarrow greedy(Q)$
$\pi \leftarrow \pi'$

# What is the problem with acting greedy?

We do not ensure we visit every possible state

Then, how can we be sure our policy is the best one?

# $\varepsilon$-greedy

**How do we choose the value of $\varepsilon$?**

Select most likely the best action, but give space for **exploration.**

$\varepsilon$ – probability to take other action
1-$\varepsilon$ – probability to act greedy


To construct a policy that is $\varepsilon$-greedy with respect to Q:

*The probability of taking action a in state s*

$$\pi(a|s) = \begin{cases} 1 - \varepsilon + \dfrac{\varepsilon}{|\mathcal{A}(s)|} & if\ a = arg\max_{a'} Q(s, a') \\ \dfrac{\varepsilon}{|\mathcal{A}(s)|} & otherwise \end{cases}$$

*Maximum action is given the remaining bulk of the probability*

*All non maximum actions are given the minimal probability of selection*

# **G**reedy in the **L**imit with **I**nfinite **E**xploration

To assure convergence to the optimal policy, 2 conditions have to be met:

1 – Every $(s, a)$ pair is visited infinitely many time
2 – The policy converges to a policy that is greedy *w.r.t.* Q

To satisfy the condition, the most common schedule for $\varepsilon$ is $\varepsilon = \frac{1}{\#episode}$

1 – $\varepsilon$ allways positive
2 – $\varepsilon$ decays to zero

# What can be improved in this method?

Right now we are improving the table after all the episodes

We need a way to update the values after **each episode**

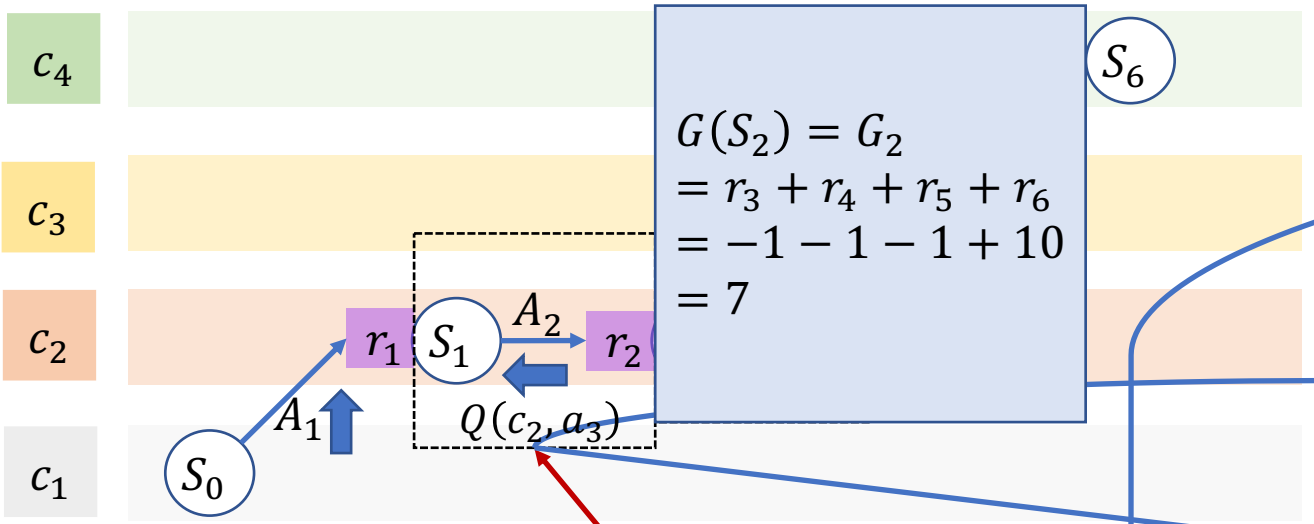
# Monte Carlo Methods

## Monte Carlo Control

Incremental updates

# Monte Carlo Control

**How do we update the table?** Suppose episode X




$Q(s,a)$

| | $a_1$ ↑ | $a_2$ ↓ | $a_3$ ← | $a_4$ → |
|---|---|---|---|---|
| $c_1$ | +5 | +4 | +2 | +4 |
| $c_2$ | +7 | +5 | ~~+8~~ + 7.8 | +9 |
| $c_3$ | +10 | +7 | +8 | +8 |

$N(s,a)$

| | $a_1$ ↑ | $a_2$ ↓ | $a_3$ ← | $a_4$ → |
|---|---|---|---|---|
| $c_1$ | +3 | +4 | +5 | +4 |
| $c_2$ | +4 | +3 | ~~+1~~ +1=+5 | +4 |
| $c_3$ | +2 | +2 | +1 | +2 |

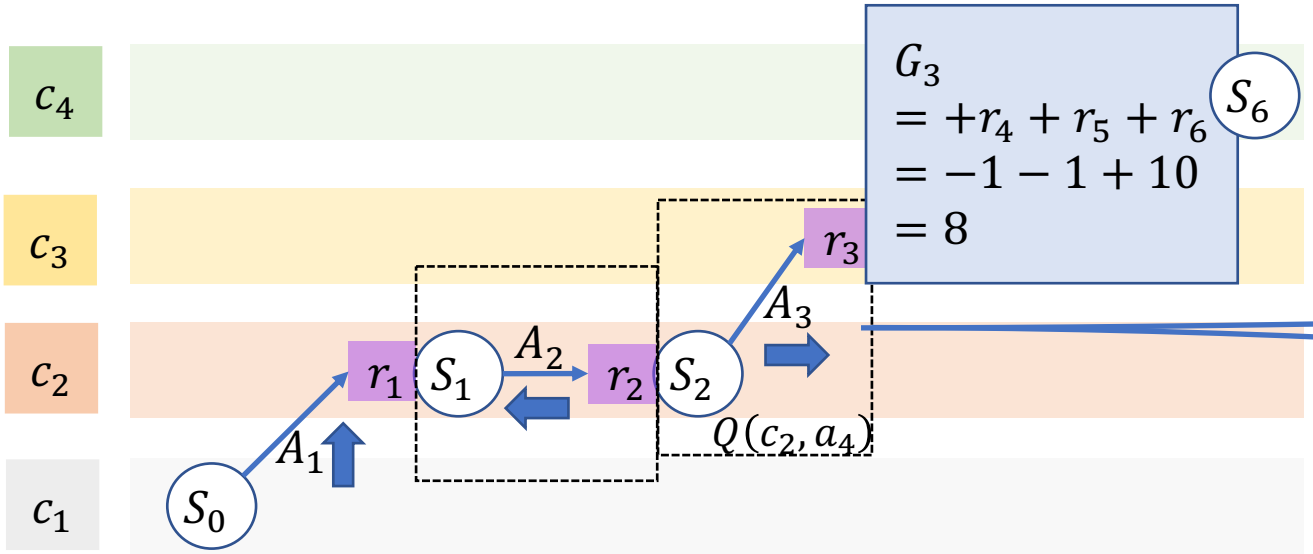
**Update Rule:** $Q(c_2, a_3) = Q(c_2, a_3) + \frac{1}{N(c_2,a_3)}\left(G_2 - Q(c_2, a_3)\right) =$

$8 + \frac{1}{5}(7 - 8) = 7.8$

**General Update Rule:** $Q(s,a) = Q(s,a) + \frac{1}{N(s,a)}\left(G_t - Q(s,a)\right)$ = Current + step size * (expected – current)

# Monte Carlo Control

**How do we update the table?** Suppose episode X




$Q(s,a)$

| | $a_1$ ↑ | $a_2$ ↓ | $a_3$ ← | $a_4$ → |
|---|---|---|---|---|
| $c_1$ | +5 | +4 | +2 | +4 |
| $c_2$ | +7 | +5 | +7.8 | +8.8 |
| $c_3$ | +10 | +7 | +8 | +8 |

$N(s,a)$

| | $a_1$ ↑ | $a_2$ ↓ | $a_3$ ← | $a_4$ → |
|---|---|---|---|---|
| $c_1$ | +3 | +4 | +5 | +4 |
| $c_2$ | +4 | +3 | +5 | +5 |
| $c_3$ | +2 | +2 | +1 | +2 |

$$Q(c_2, a_4) = Q(c_2, a_4) + \frac{1}{N(c_2, a_4)}\left(G_2 - Q(c_2, a_4)\right) = 9 + \frac{1}{5}(8 - 9) = 8.8$$

# What can be improved in this method?

$\frac{1}{N(s,a)}$ will be high at the beginning and get smaller with time

This is undesired, we want last information to be indeed more important

# Monte Carlo Methods

## Monte Carlo Control

### Constant Step Size $\alpha$

**General Update Rule:** $Q(s,a) = Q(s,a) + \frac{1}{N(s,a)}\big(G_t - Q(s,a)\big)$ = Current + step size * (expected – current)

**General Update Rule:** $Q(s,a) = Q(s,a) + \alpha\big(G_t - Q(s,a)\big)$ = Current + fix step size * (expected – current)

Coding example of Monte Carlo Prediction playing Blackjack

Coding example of Monte Carlo Control playing Blackjack

**Algorithm 11:** First-Visit Constant-$\alpha$ (GLIE) MC Control

**Input:** positive integer *num_episodes*, small positive fraction $\alpha$, GLIE $\{\epsilon_i\}$
**Output:** policy $\pi$ ($\approx \pi_*$ if *num_episodes* is large enough)
Initialize $Q$ arbitrarily (e.g., $Q(s,a) = 0$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$)
**for** $i \leftarrow 1$ ***to*** *num_episodes* **do**
  $\epsilon \leftarrow \epsilon_i$
  $\pi \leftarrow \epsilon$-greedy$(Q)$
  Generate an episode $S_0, A_0, R_1, \ldots, S_T$ using $\pi$
  **for** $t \leftarrow 0$ ***to*** $T-1$ **do**
    **if** $(S_t, A_t)$ *is a first visit (with return* $G_t$*)* **then**
      $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(G_t - Q(S_t, A_t))$
  **end**
**end**
**return** $\pi$

# Recap of Monte Carlo Algorithms

**Algorithm 9:** First-Visit MC Prediction (*for action values*)

**Input:** policy $\pi$, positive integer *num_episodes*
**Output:** value function $Q$ ($\approx q_\pi$ if *num_episodes* is large enough)
Initialize $N(s,a) = 0$ for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$
Initialize $returns\_sum(s,a) = 0$ for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$
**for** $i \leftarrow 1$ ***to*** *num_episodes* **do**
  Generate an episode $S_0, A_0, R_1, \ldots, S_T$ using $\pi$
  **for** $t \leftarrow 0$ ***to*** $T-1$ **do**
    **if** $(S_t, A_t)$ *is a first visit (with return* $G_t$*)* **then**
      $N(S_t, A_t) \leftarrow N(S_t, A_t) + 1$
      $returns\_sum(S_t, A_t) \leftarrow returns\_sum(S_t, A_t) + G_t$
  **end**
**end**
$Q(s,a) \leftarrow returns\_sum(s,a)/N(s,a)$ for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$
**return** $Q$

**Algorithm 10:** First-Visit GLIE MC Control

**Input:** positive integer *num_episodes*, GLIE $\{\epsilon_i\}$
**Output:** policy $\pi$ ($\approx \pi_*$ if *num_episodes* is large enough)
Initialize $Q(s,a) = 0$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$
Initialize $N(s,a) = 0$ for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$
**for** $i \leftarrow 1$ ***to*** *num_episodes* **do**
  $\epsilon \leftarrow \epsilon_i$
  $\pi \leftarrow \epsilon$-greedy$(Q)$
  Generate an episode $S_0, A_0, R_1, \ldots, S_T$ using $\pi$
  **for** $t \leftarrow 0$ ***to*** $T-1$ **do**
    **if** $(S_t, A_t)$ *is a first visit (with return* $G_t$*)* **then**
      $N(S_t, A_t) \leftarrow N(S_t, A_t) + 1$
      $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{1}{N(S_t, A_t)}(G_t - Q(S_t, A_t))$
  **end**
**end**
**return** $\pi$

**Algorithm 11:** First-Visit Constant-$\alpha$ (GLIE) MC Control

**Input:** positive integer *num_episodes*, small positive fraction $\alpha$, GLIE $\{\epsilon_i\}$
**Output:** policy $\pi$ ($\approx \pi_*$ if *num_episodes* is large enough)
Initialize $Q$ arbitrarily (e.g., $Q(s,a) = 0$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$)
**for** $i \leftarrow 1$ ***to*** *num_episodes* **do**
  $\epsilon \leftarrow \epsilon_i$
  $\pi \leftarrow \epsilon$-greedy$(Q)$
  Generate an episode $S_0, A_0, R_1, \ldots, S_T$ using $\pi$
  **for** $t \leftarrow 0$ ***to*** $T-1$ **do**
    **if** $(S_t, A_t)$ *is a first visit (with return* $G_t$*)* **then**
      $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(G_t - Q(S_t, A_t))$
  **end**
**end**
**return** $\pi$