Montecarlo

Montecarlo

Programación dinámica

$$\{S, A, R, p, \gamma\}$$

Monte Carlo

$$\{S, A, R, p, \gamma\}$$

No existe modelo del entorno

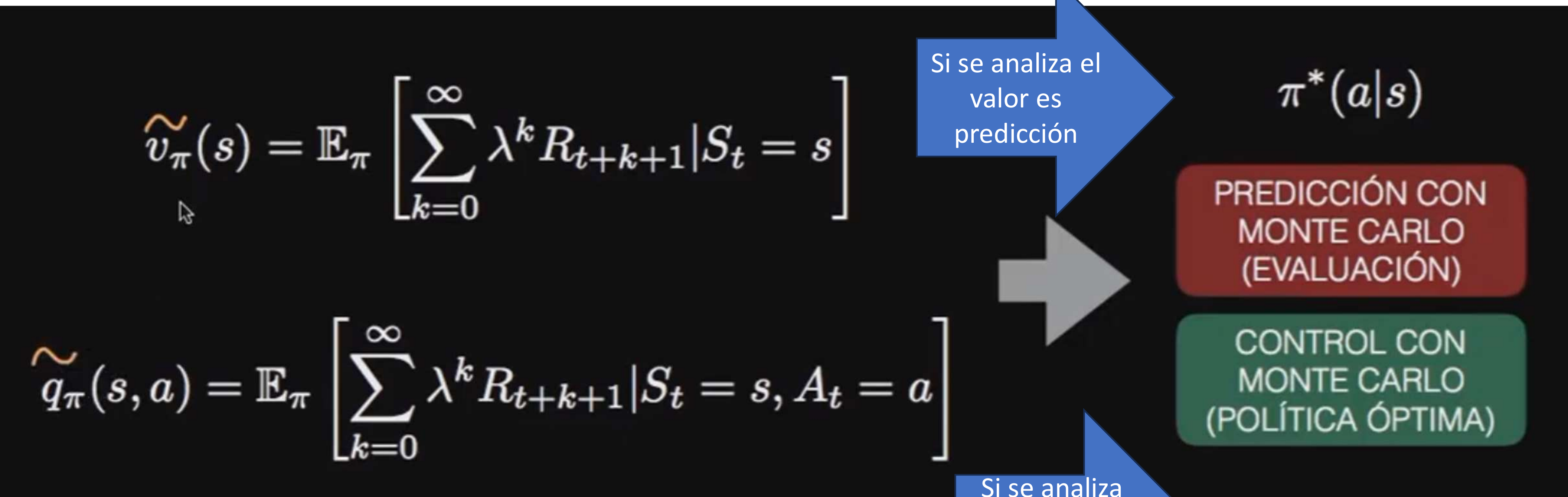
Montecarlo

$$v_\pi(s) = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \lambda^k R_{t+k+1} | S_t = s \right]$$

Estimación función estado valor

Promedio

Múltiples trayectorias (Iteraciones)



Si se analiza el valor es predicción

Si se analiza la función acción valor será control

Montecarlo

- Los métodos de Monte Carlo en aprendizaje por refuerzo son algoritmos que **estiman las funciones de valor** al recopilar episodios completos de interacción con el entorno y calcular recompensas acumuladas basadas en las trayectorias observadas

Montecarlo

En el contexto de un **mundo grilla** (gridworld), estos métodos permiten evaluar estados y políticas mediante:

1. **Promedio de retornos**: Calculan el valor de cada estado como el promedio de las recompensas acumuladas a largo plazo obtenidas al visitarlo.

2. **Muestreo directo**: No requieren un modelo del entorno, sino únicamente muestras de episodios completos. (Como las probabilidades de transición entre estados ni las recompensas asociadas a dichas transiciones)

3. **Evaluación de políticas**: Comparan políticas mejorando iterativamente la acción elegida en cada estado según los retornos observados.

Montecarlo

- Los métodos de Monte Carlo en RL se dividen en dos partes, al igual que la iteración de políticas: **Predicción de MC** y **Control de MC.**

Predicción Montecarlo

Predicción MC

- De manera muy similar a cómo evaluamos la función de valor para la evaluación de políticas, nos centraremos en evaluar la **función q** en la predicción de MC.

- Nuevamente, la pregunta será: "dada una política, ¿cuál es la función q para esa política correspondiente?"

$$q_\pi(s,a) = \mathbb{E}\left[\sum_{k=0}^{T} \gamma^k R_{t+k+1} \mid S_t = s, A_t = a\right]$$

Predicción MC

$$q_\pi(s,a) = \mathbb{E}\left[\sum_{k=0}^{T} \gamma^k R_{t+k+1} \mid S_t = s, A_t = a\right]$$

- La función q es básicamente la **suma de las recompensas esperadas** en un episodio hasta el paso de tiempo terminal dado el par estado-acción actual (s, a) siguiendo la política **π**.

Predicción MC

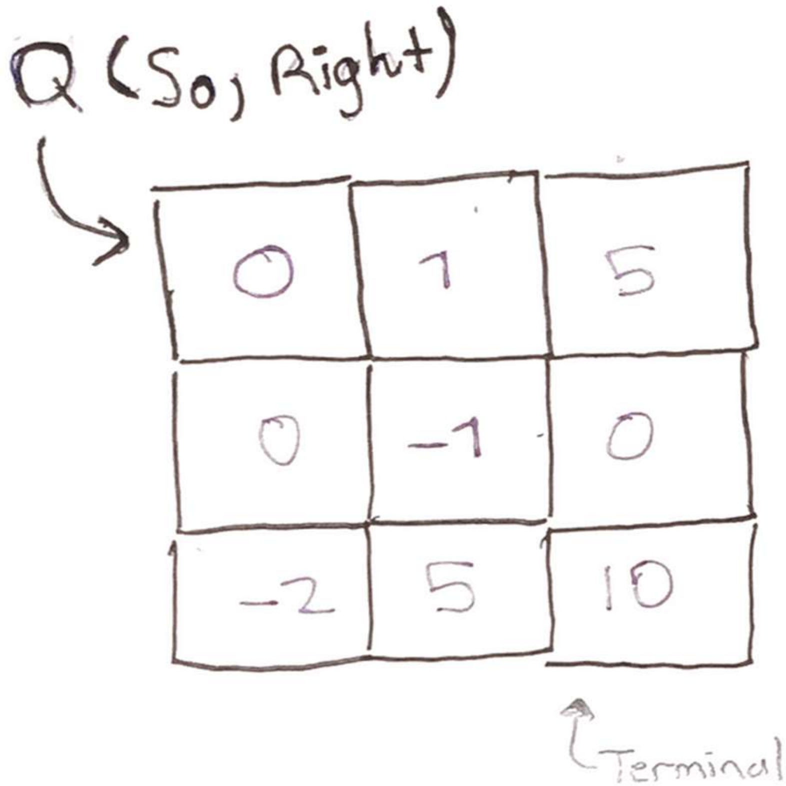
- Sin embargo, como no conocemos la función de transición de estado para el entorno en MC, no nos es posible calcular la expectativa ya que **no conocemos las probabilidades** asociadas con cada recompensa.

$$q_\pi(s,a) = \mathbb{E}\left[\sum_{k=0}^{T} \gamma^k R_{t+k+1} \mid S_t = s, A_t = a\right]$$

Predicción MC

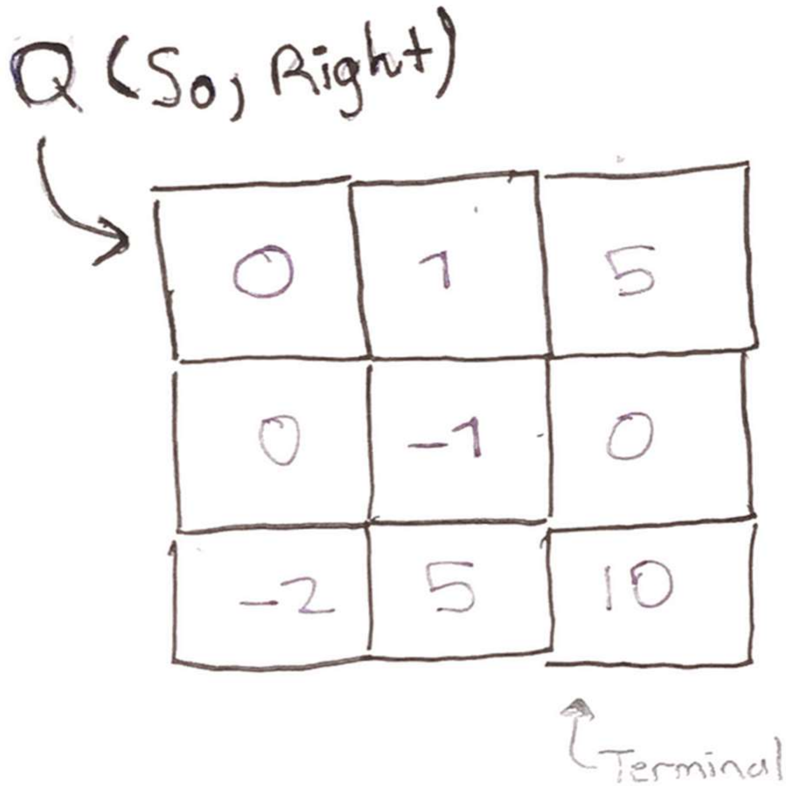
- Sin embargo, lo que podemos hacer es **ejecutar una serie de episodios con acciones aleatorias** y observar las recompensas recibidas en cada episodio.

Predicción MC



- Los valores representan las recompensas por la transición a ese estado.
- El estado con +10 recompensas es el estado terminal

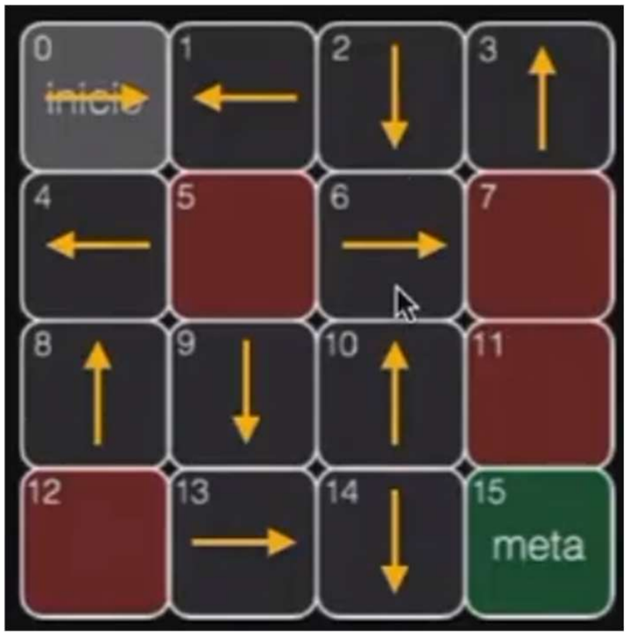
Predicción MC



- Dado este pequeño mundo cuadrícula, nuestro objetivo es calcular el valor q de estar en el estado S_0 y tomar la acción correcta.
- Recordemos que en los métodos de MC tenemos que promediar las recompensas.
- Esto significa que tenemos que calcular la media empírica en lugar de la expectativa, ya que no conocemos las probabilidades.

Ejemplo

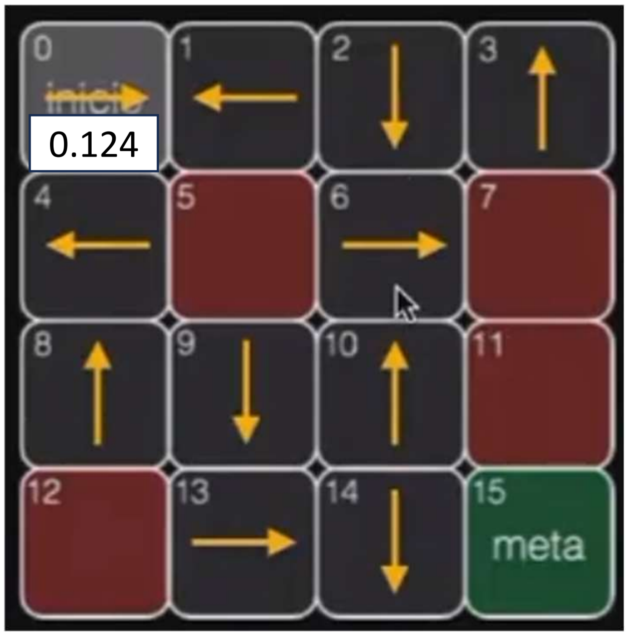
Ejemplo



$$G = \left[\sum_{k=0}^{\infty} \lambda^k R_{t+k+1} | S_t = s\right], \gamma = 0.9$$

Episodio	Trayectoria	Retorno (G)
1	0 → 1 → 2 → 6 → 7	0
2	0 → 1 → 2 → 6 → 5	0
3	0 → 1 → 2 → 6 → 10 → 9 → 13 → 14 → 15	0.43
4	0 → 4 → 5	0
5	0 → 4 → 8 → 9 → 10 → 14 → 15	0.53
6	0 → 4 → 8 → 12	0
7	0 → 1 → 2 → 3 → 2 → 6 → 10 → 9 → 8 → 9 → 13 → 14 → 15	0.28
8	0 → 1 → 2 → 6 → 10 → 6 → 7	0
9	0 → 1 → 0 → 4 → 8 → 9 → 5	0
10	0 → 1 → 2 → 3 → 2 → 1 → 5	0

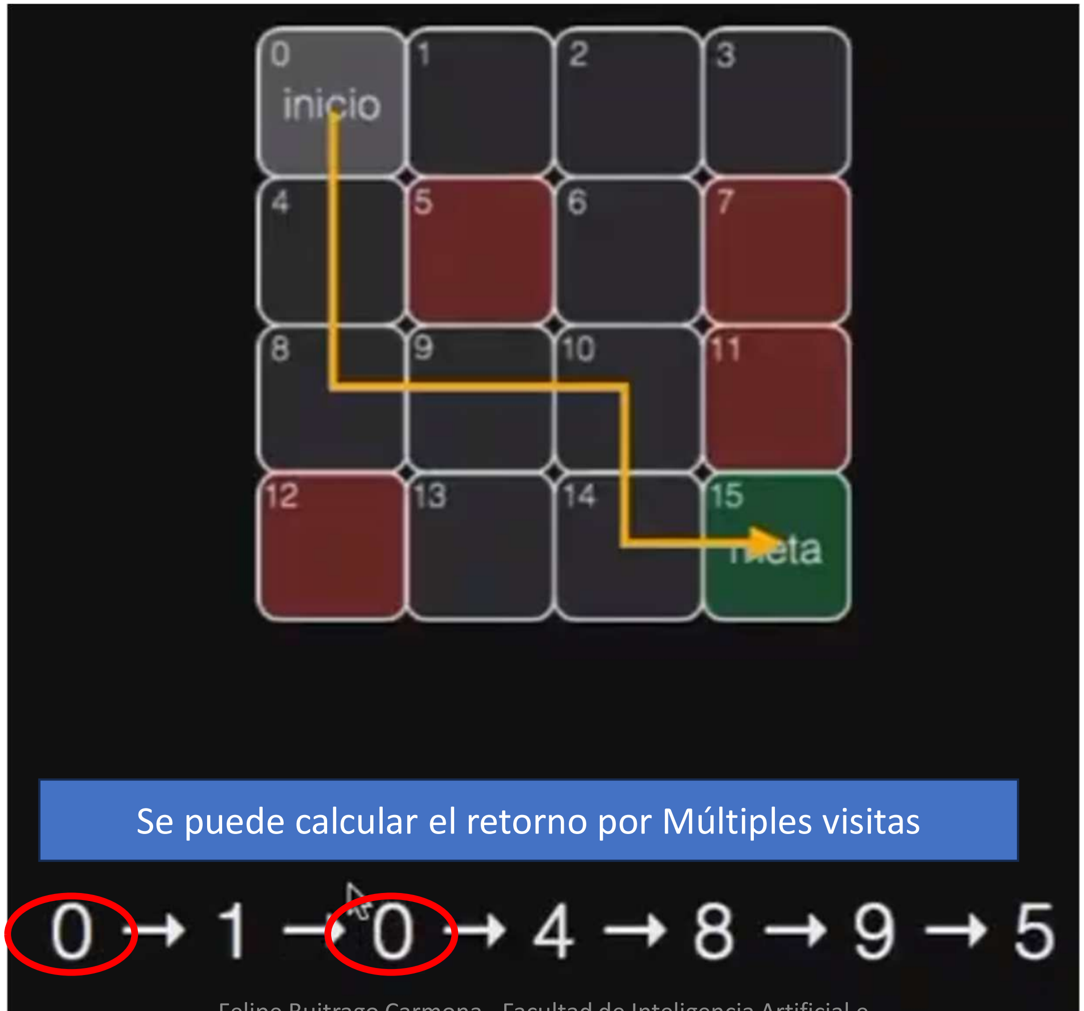
Ejemplo

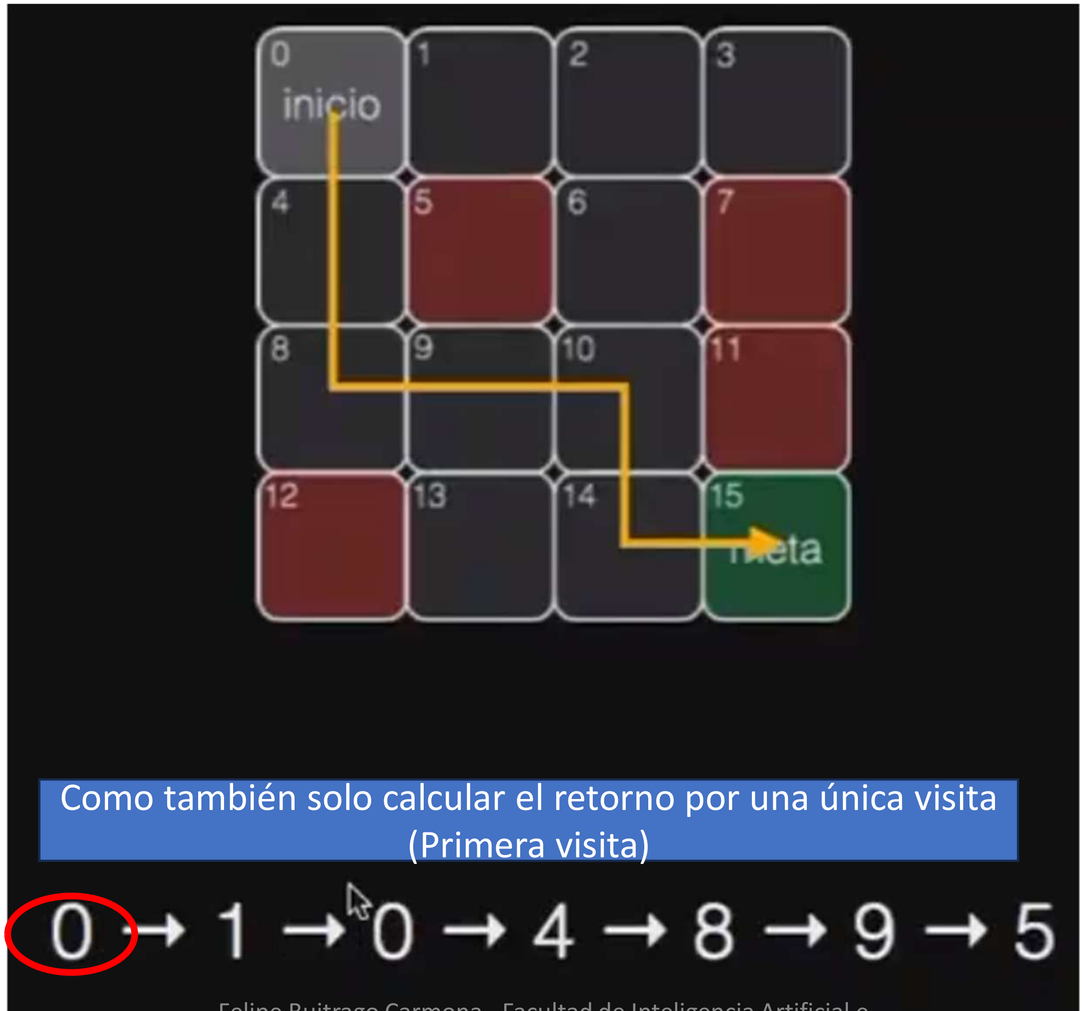


Episodio	Trayectoria	Retorno (G)
1	0 → 1 → 2 → 6 → 7	0
2	0 → 1 → 2 → 6 → 5	0
3	0 → 1 → 2 → 6 → 10 → 9 → 13 → 14 → 15	0.43
4	0 → 4 → 5	0
5	0 → 4 → 8 → 9 → 10 → 14 → 15	0.53
6	0 → 4 → 8 → 12	0
7	0 → 1 → 2 → 3 → 2 → 6 → 10 → 9 → 8 → 9 → 13 → 14 → 15	0.28
8	0 → 1 → 2 → 6 → 10 → 6 → 7	0
9	0 → 1 → 0 → 4 → 8 → 9 → 5	0
10	0 → 1 → 2 → 3 → 2 → 1 → 5	0

$$\widetilde{v_\pi}(s=0) = \frac{0+0+0+0.43+0+0.53+0+0.28+0+0+0}{10} = 0.124$$

Eso quiere decir que el valor del estado cero será de 0.124. Entre mas iteraciones mejor la aproximación. Esto se debe realizar por cada estado


0
inicio
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Se puede calcular el retorno por Múltiples visitas
0 → 1 → 0 → 4 → 8 → 9 → 5



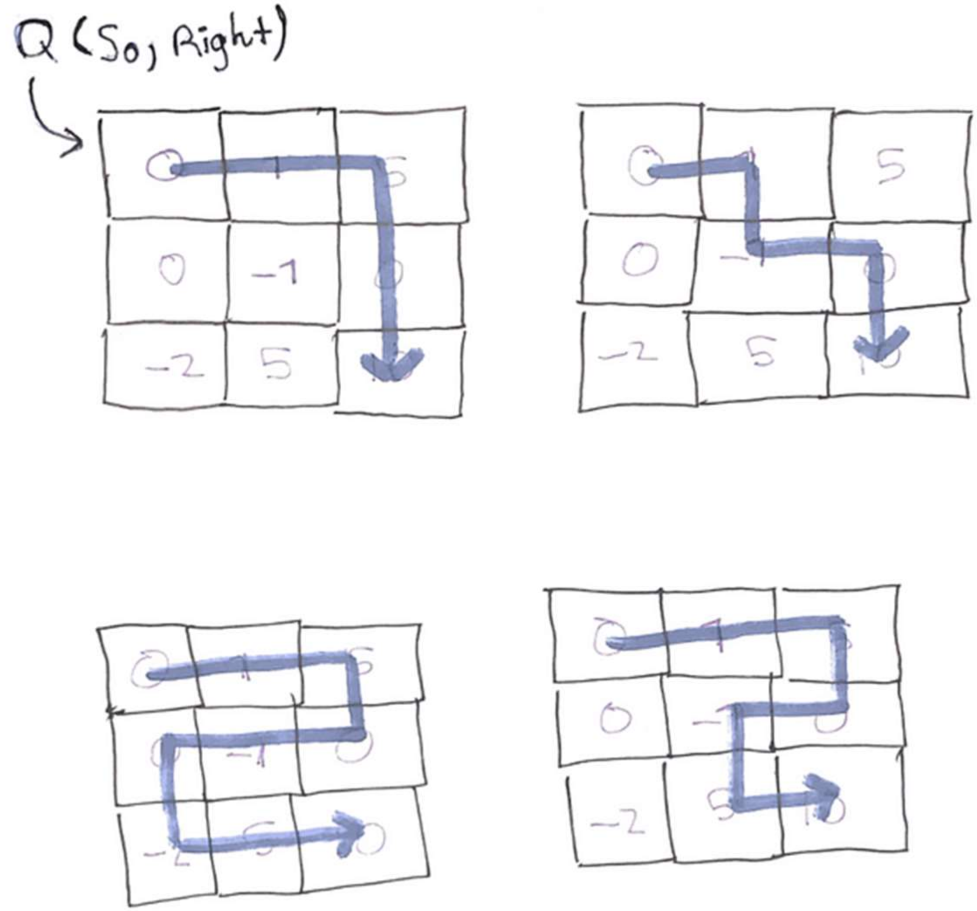
Como también solo calcular el retorno por una única visita (Primera visita)

- Entrada:
  - $\pi \leftarrow$ la política a evaluar
- Inicializar:
  - $V(s) \leftarrow$ valores arbitrarios para cada estado "s" (que se actualizará con el algoritmo)
  - Retornos(s) $\leftarrow$ lista vacía (que contendrá el valor del retorno obtenido para cada estado "s")
- Repetir K veces y en cada iteración:
  - (a) Generar un episodio usando π
  - (b) $G \leftarrow 0$ (inicializar el retorno a calcular)
  - (c) Por cada instante de tiempo "t" en el episodio:
    - $G \leftarrow \gamma G + R_{t+1}$ (actualizar el retorno)
    - Si es la primera visita a "s":
      - Retornos(s) $\leftarrow$ [Retornos(s), G] (agregar el retorno calculado)
      - $V(St) \leftarrow$ promedio(Retornos(s)) (y promediar los retornos para obtener la función estado-valor)

Ejemplo

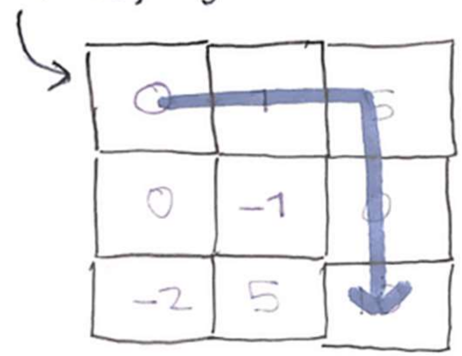
Ejemplo

- Primero, inicializaremos todos nuestros valores q en 0 y estableceremos una política estocástica aleatoria **π**. Reproduciremos 4 episodios y acumularemos 4 retornos. Ahora será fácil calcular la media incremental. El factor de descuento se toma como 1 para simplificar la explicación.



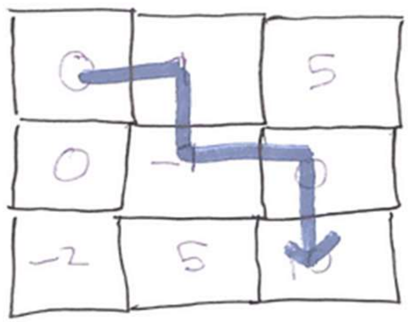
Ejemplo

Q (S0, Right)



Ejemplo de retorno 1:

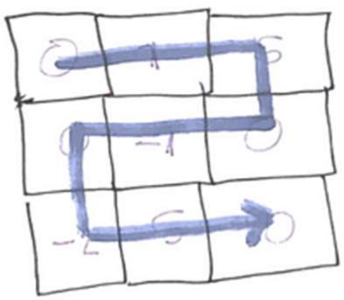
G_t = 0 + 1 + 5 + 0 + 10 = 16



Ejemplo de retorno 2:

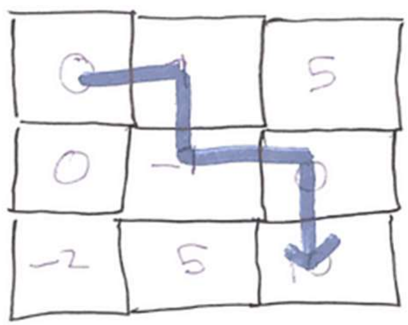
G_t = 0 + 1 - 1 + 0 + 10 = 10

$$q_\pi(s_0, \text{right}) = \frac{16 + 10 + 18 + 20}{4} = 16$$



Ejemplo de retorno 3:

G_t = 0 + 1 + 5 + 0 - 1 + 0 - 2 + 5 + 10 = 18



Ejemplo de retorno 4:

G_t = 0 + 1 + 5 + 0 - 1 + 5 + 10 = 20

Ejemplo

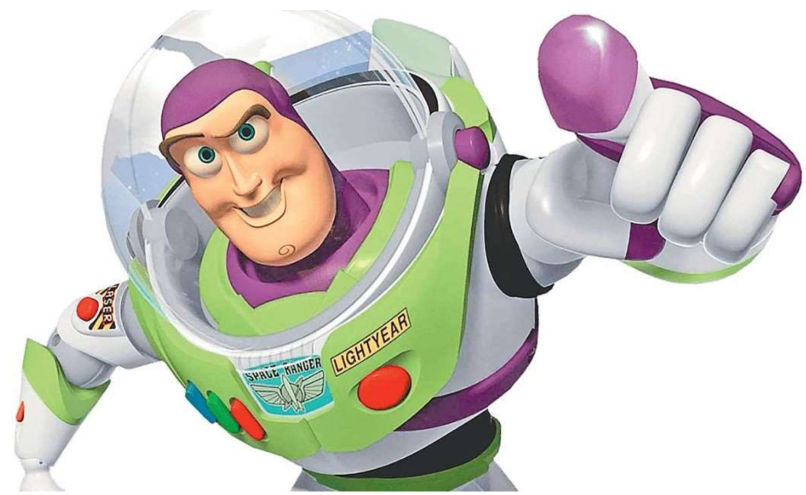
- Al utilizar la media incremental, podemos estimar el valor q de estar en el estado S_0 y tomar la acción correcta. Pero surge una pregunta: ¿cómo nos garantizará el cálculo de la media empírica que este valor q será la aproximación precisa de la función q para la política dada?

$$q_\pi(\mathrm{s}_0, \mathrm{right}) = \frac{16 + 10 + 18 + 20}{4} = 16$$

Ejemplo

- Según la ley de los grandes números, a medida que aumenta el tamaño de una muestra, su media se acerca al promedio de toda la población.

- En la predicción de MC, a medida que el número de muestras devueltas se acerca al infinito, el promedio de todas esas muestras devueltas será cada vez más una estimación precisa del valor q.



Montecarlo

- **¡Ésta es la esencia de los Métodos de Monte Carlo!**
- Pudimos evaluar la función q_function SIN tener que conocer la dinámica del entorno.

Montecarlo

- Otra forma de encontrar el valor medio de los rendimientos totales es actualizar el incremento medio utilizando la siguiente regla de actualización:

$$Q \leftarrow Q + \frac{1}{N}(G - Q)$$

•Q representa el valor q para un par estado-acción

•G es el rendimiento actual acumulado después de que el episodio ha terminado

•N es el número total de veces que se ha encontrado ese par estado-acción

•G - Q se denomina **diferencia temporal** , ya que estamos restando el antiguo rendimiento promedio (Q) del nuevo rendimiento (G).

- Es importante recordar que **los valores Q se actualizan después de que finaliza un episodio en Monte Carlo Learning**

Montecarlo

- Otra cosa interesante que cabe destacar es que una vez que el valor de N se vuelve relativamente grande, la diferencia temporal no tendrá mucho efecto en la actualización del valor Q.
- Esto es un problema porque las actualizaciones en los episodios anteriores se verán favorecidas más que las actualizaciones en los episodios posteriores, lo que hace que nuestro algoritmo esté sesgado.
- Para combatir esta noción, multiplicaremos por un alfa constante en lugar de N

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \boxed{\alpha(G_t - Q(S_t, A_t))}$$

Nota (Opcional)

- Si está familiarizado con el descenso de gradiente del aprendizaje automático, esta regla de actualización se parece mucho al cálculo de los pesos.

- Tiene su antiguo valor Q, la "pérdida" cuantificada como G - Q, y multiplica esa pérdida por una constante. La única diferencia es que sumamos la pérdida en lugar de restarla.

- Al sumar la pérdida, podemos acercarnos a la media debido a la naturaleza incremental del cálculo de la media.

Montecarlo

- Ahora que podemos calcular el valor q para cada par estado-acción, todo lo que tenemos que hacer es ejecutar muchos episodios hasta que la función q aleatoria se convierta en la verdadera función q para la política **π**

$$\mathrm{q}_k \approx q_\pi \ \text{ as } \ k \to \infty$$

Control Montecarlo

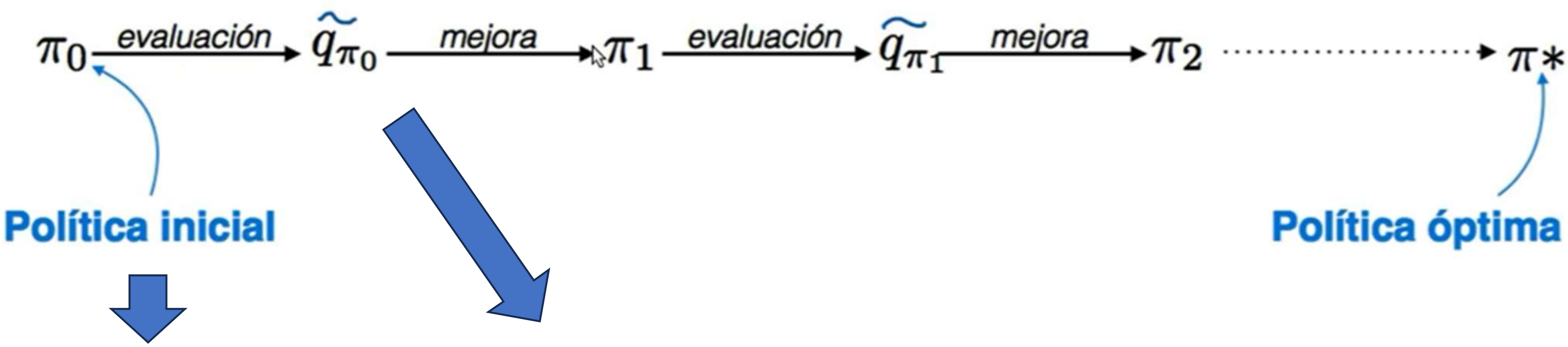
Control Montecarlo

- Ahora que hemos calculado nuestra función q, tenemos que encontrar la mejor política que maximice nuestros valores q para cada estado.

- El control de MC es básicamente una mejora de la política, pero con una diferencia sutil: **tenemos que lidiar con el problema de exploración y explotación.**

Control Montecarlo

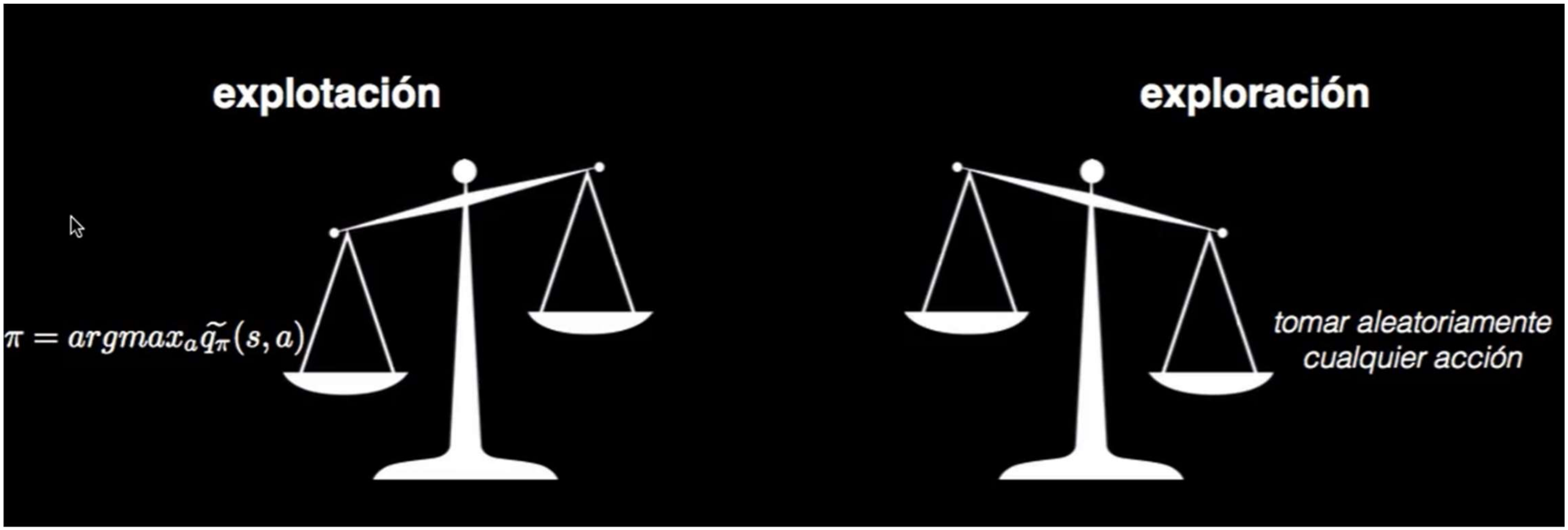
- Objetivo: Obtener una política óptima combinando iterativamente algoritmos de evaluación y mejora



Inicialmente predicción montecarlo

Estimación función acción valor

Explotación VS Exploración



Explotación

Elige solo la acción que devuelva el valor q más alto

Exploración

Elegir acciones al azar de todo el espacio de acción

Control Montecarlo

- Al igual que en la programación dinámica, vamos a elegir la acción con ese valor q máximo en un estado determinado. Sin embargo, también tenemos que tener en cuenta las trayectorias que nuestro agente aún no ha tomado.

- Ejemplo (Leer en 3 minutos):

  Si un agente encuentra una acción a_1 del estado S_0 y recibe un retorno de +5, tendría sentido que ese agente explotara esa acción dado que sería el valor q más alto para ese estado dado. Sin embargo, ¿qué sucede si hay otra acción a_2 del estado S_0 que dará un retorno aún mejor de +10?

Ejemplo

Control Montecarlo

- Ejemplo (Leer en 3 minutos):

  Si un agente encuentra una acción a_1 del estado S_0 y recibe un retorno de +5, tendría sentido que ese agente explotara esa acción dado que sería el valor q más alto para ese estado dado. Sin embargo, ¿qué sucede si hay otra acción a_2 del estado S_0 que dará un retorno aún mejor de +10?

  Si nuestro agente siempre decidiera elegir la acción con el valor q más alto (en este caso sería +5 ya que nuestro agente no ha explorado todas las demás acciones), ¡en realidad estaría eligiendo la acción subóptima cada vez!

Control Montecarlo

- Por esta razón, necesitamos idear algún método donde el agente elija la acción subóptima con cierta probabilidad de tiempo para garantizar que explore todas las acciones posibles en un estado dado.

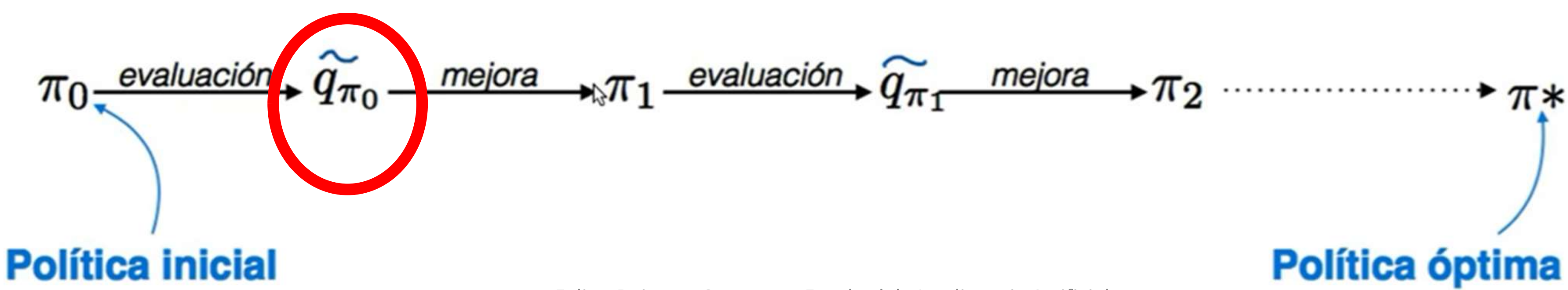
Control Montecarlo
Estrategia Epsilon-Codiciosa

Control Montecarlo
Estrategia Epsilon-Codiciosa

- Una de las formas de abordar este problema de exploración es mediante la estrategia epsilon greedy.
- Sea ε la probabilidad de que vayamos a realizar una acción aleatoria entre 0.1 y 0.2
- Esto significa que vamos a elegir la acción más óptima con una probabilidad de $1 - \varepsilon$ en ese momento.

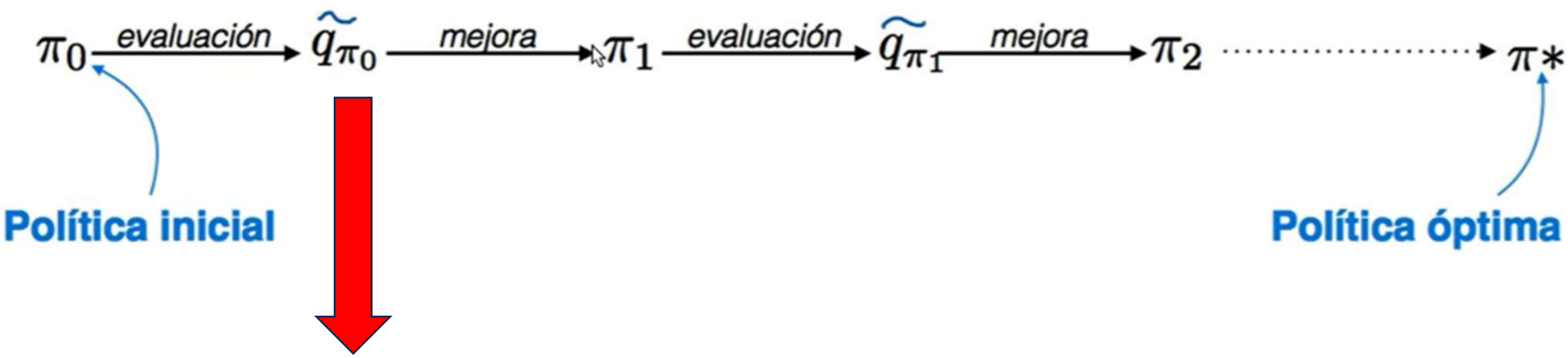
Control Montecarlo
Estrategia Epsilon-Codiciosa

- Una vez se obtenga la función acción valor, se debe seleccionar la mejor acción posible (para generar una nueva política)
- No se va a elegir la mejor acción posible (como en mejora de política [porque no se puede visitar todo] y tampoco aleatoria)
- Debe ser un punto intermedio



Control Montecarlo
Estrategia Epsilon-Codiciosa

- Punto intermedio



Probabilidad de 1 – ε → Acción que maximiza la función $\arg\max(\mathrm{a}) q_{\pi}(s, a)$ → Explotación
Probabilidad de ε → Acción aleatoria → Exploración

Control Montecarlo
Estrategia Epsilon-Codiciosa - On Policy

- Parámetro del algoritmo: ϵ (entre 0 y 1)
- Inicializar:
  - $\pi \leftarrow$ probabilidades aleatorias (la política a optimizar)
  - $Q(s,a) \leftarrow$ valores arbitrarios para cada estado "s" y acción "a" (que se actualizará con el algoritmo)
  - Retornos(s,a) $\leftarrow$ lista vacía (que contendrá el valor del retorno obtenido para cada estado "s" y acción "a")
- Repetir K veces y en cada iteración:
  (a) Generar un episodio usando π
  (b) $G \leftarrow 0$ (inicializar el retorno a calcular)
  (c) Por cada instante de tiempo "t" en el episodio:
  - $G \leftarrow \gamma G + R_{t+1}$ (actualizar el retorno)
  - Si es la primera visita a S_t, A_t:
    - Retornos(S_t, A_t) $\leftarrow$ [Retornos(S_t, A_t), G] (agregar el retorno calculado)
    - $Q(S_t, A_t) \leftarrow$ promedio(Retornos(S_t, A_t)) (actualizar la estimación de la función acción-valor)
    - $A^* \leftarrow \operatorname{argmax}_a Q(S_t, a)$ (obtener la acción que maximiza la función acción-valor
    - Mejorar la política: para cada acción "a":
      - $\pi(a|St) = 1 - \epsilon + \epsilon/m$, si $a = A^*$
      - $\pi(a|St) = \epsilon/m$, si $a \neq A^*$

Evaluación

Mejora

Política sub óptima, porque estamos generando espacio para decisiones aleatorias. Eso quiere decir que podemos llegar a una política que no es la mejor

Control Montecarlo
Estrategia Epsilon-Codiciosa

On-Policy

Repetir K veces y en cada iteración:

(a) Generar un episodio usando π

(b) $G \leftarrow 0$ (inicializar el retorno a calcular)

(c) Por cada instante de tiempo "t" en el episodio

- $G \leftarrow \gamma G + R_{t+1}$ (actualizar el retorno)
- Si es la primera visita a S_t, A_t:
  - Retornos(S_t, A_t) $\leftarrow$ [Retornos(S_t, A_t), G] (
  - $Q(S_t, A_t) \leftarrow$ promedio(Retornos(S_t, A_t)) (a
  - $A^* \leftarrow \text{argmax}_a\, Q(S_t, a)$ (obtener la acción
  - Mejorar la política: para cada acción "a":
    - $\pi(a|St) = 1 - \epsilon + \epsilon/m, \quad$ si $a = A^*$
    - $\pi(a|St) = \epsilon/m, \quad$ si $a \neq A^*$

Porque la política que se está utilizando es la misma que se está utilizando para generar los episodios y para realizar las mejoras

Control Montecarlo
Estrategia Epsilon-Codiciosa

Off-Policy

Crear una política “b” para definir las interacciones

Política pi que es la que se quiere optimizar

Se llama Off Policy porque la política que se está optimizando no es la misma que se está usando para controlar las acciones del agente en el entorno

Control Montecarlo
Estrategia Epsilon-Codiciosa

- Como pasar de tener una política b obtener una política óptima pi?

- Se requiere calcular una métrica de semejanza o diferencia entre las 2

$$\pi_0 \xrightarrow{\text{evaluación}} \widetilde{q}_{\pi_0} \xrightarrow{\text{mejora}} \pi_1 \xrightarrow{\text{evaluación}} \widetilde{q}_{\pi_1} \xrightarrow{\text{mejora}} \pi_2 \cdots\cdots\cdots \rightarrow \pi*$$

Política inicial

$$¿\widetilde{q}_b(s, a) \longrightarrow \widetilde{q}_\pi(s, a)?$$

Política óptima

Control Montecarlo
Estrategia Epsilon-Codiciosa

proporción de muestreo de importancia (*importance-sample ratio*)

$$\rho = \frac{Pr\{A_t, S_{t+1}, A_{t+1}, \ldots, S_T | S_t, A_t \backsim \pi\}}{Pr\{A_t, S_{t+1}, A_{t+1}, \ldots, S_T | S_t, A_t \backsim b\}} = \prod_{k=t}^{T-1} \frac{\pi(A_k|S_k)}{b(A_k|S_k)}$$

- Probabilidad de generar una trayectoria determinada usando la política pi con la probabilidad de generar esa misma trayectoria con la política b.
- No es necesario tener en cuenta los otros elementos de Marcov

Control Montecarlo
Estrategia Epsilon-Codiciosa

- El valor anteriormente mencionado puede ser usado para lo siguiente:

- De la política b se puede calcular una ganancia y al multiplicar este valor por ρ se puede obtener la ganancia de la política π

- $G_\pi = \rho * G_b$ $$¿\widetilde{q}_b(s,a) \longrightarrow \widetilde{q}_\pi(s,a)?$$

- Por tanto: $$\widetilde{q}_\pi(s,a) = E_\pi[\rho * G_b | S_t = s, A_t = a]$$

Control Montecarlo
Estrategia Epsilon-Codiciosa – Off Policy

π W

- Inicializar, para todos los estados "s" y todas las acciones "a":
  - $Q(s,a) \leftarrow$ valores arbitrarios para cada estado "s" y acción "a" (que se actualizará con el algoritmo)
  - $C(s,a) \leftarrow 0$ (acumulador para la suma de los valores de $\boldsymbol{\rho}$ obtenidos en las diferentes iteraciones)
  - $\pi \leftarrow$ probabilidades aleatorias (la política a optimizar)
- Repetir K veces y en cada iteración:
  (a) $b \leftarrow$ una política arbitraria
  (b) Generar un episodio usando b
  (c) $W \leftarrow 1$ (el valor de $\boldsymbol{\rho}$ que se actualizará en cada iteración)
  (d) Por cada instante de tiempo "t" en el episodio:
  - $G \leftarrow \gamma G + R_{t+1}$ (actualizar el retorno)
  - $C(S_t, A_t) \leftarrow C(S_t, A_t) + W$ (actualizar el acumulador)
  - $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + [W/C(S_t, A_t)].[G - Q(S_t, A_t)]$ (actualizar la estimación de la función acción-valor)
  - $\pi \leftarrow \text{argmax}_a\, Q(S_t, a)$ (actualizar la política de forma codiciosa)
  - $W \leftarrow W.[\pi(A_t|S_t)/b(A_t|S_t)]$