Aprendizaje Reforzado

Maestría en Data Mining, Universidad Austral

Javier Kreiner

Plan de la clase

Teoría:

- Exploración vs Explotación
- Bandidos Multi-Brazo
- 10-armed Test Bed
- Métodos: ε-greedy, optimista, cota superior de confianza
- Cadenas de Markov
- Procesos de Decisión de Markov
- Procesos de Recompensa de Markov
- Función de Valor
- Función de Acción-Valor
- Ecuación de Esperanza de Bellman
- Ecuación de Optimalidad de Bellman
- Evaluación de una política

Exploración vs Explotación

- **Explotación**: Tomar la acción que es **más conveniente** *en el momento*.
- **Exploración**: Tomar **decisiones sub-óptimas** con el propósito de *obtener más información*.

Ejemplos:

¿Qué publicidad muestro? ¿Dónde realizó pozos de petróleo? ¿Qué restorán elijo?

Bandidos Multi-brazo

Sólo hay acciones y recompensas.

$$A_t \longrightarrow R_t$$

$$q_*(a) = E\left[R_t \mid A_t = a\right]$$

$$a = 1, \dots, k$$

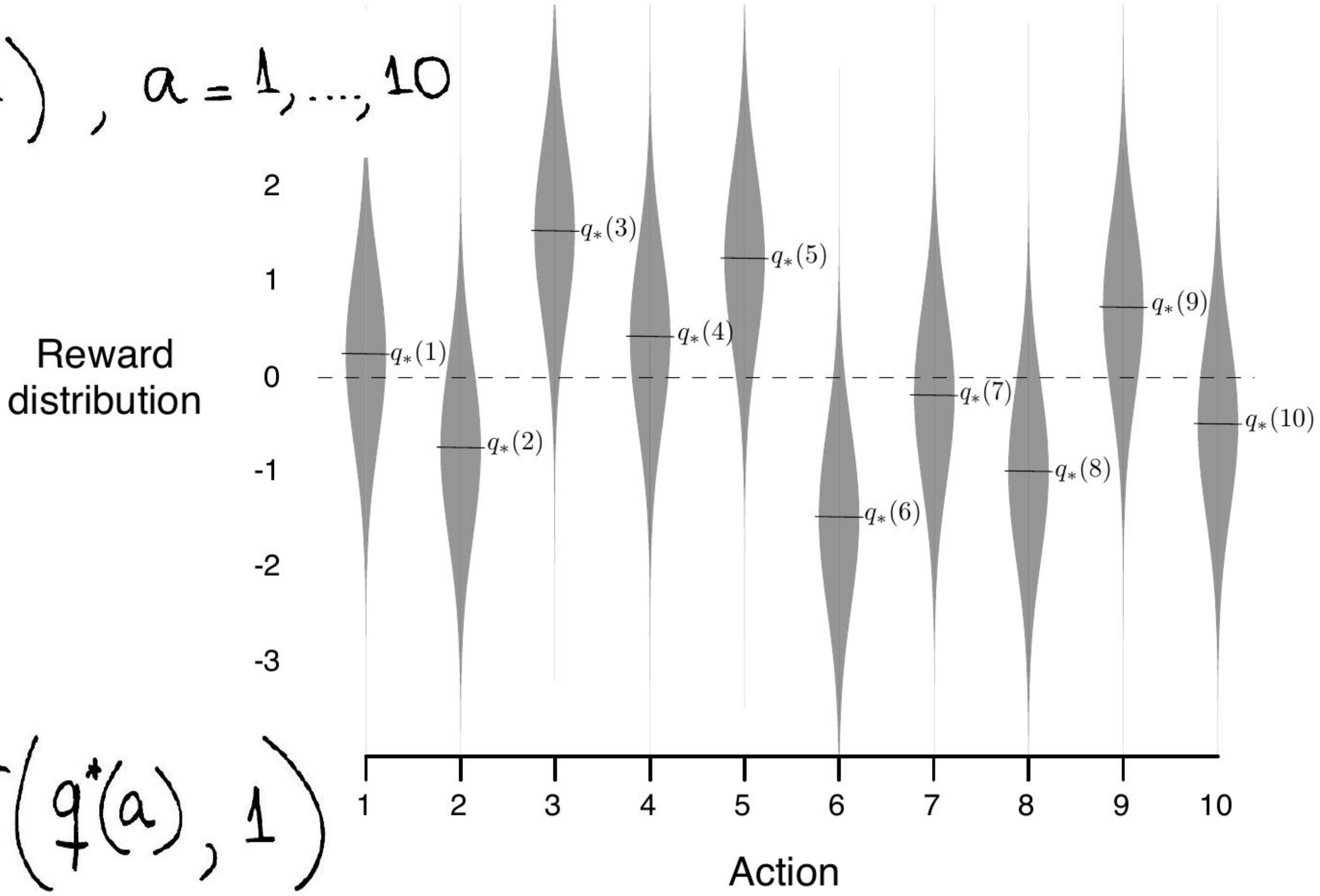
Métodos basados en la función de valor

$$Q_t(a) = \frac{\sum_{i=1}^{t-1} R_i \, \mathbb{1}_{A_i = a}}{N_t(a)}$$

$$A_t = \operatorname*{argmax}_a Q_t(a)$$

10-armed Testbed

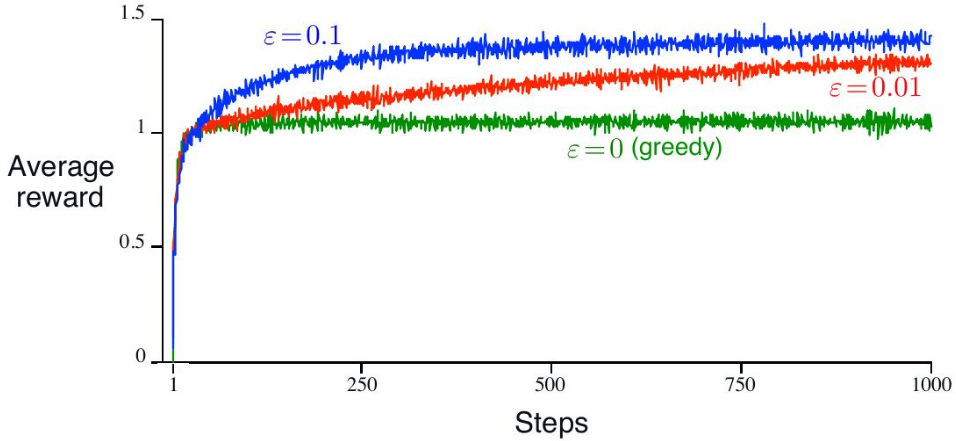
$$q^*(a) \sim \mathcal{N}(0, 1), \quad a = 1, \ldots, 10$$

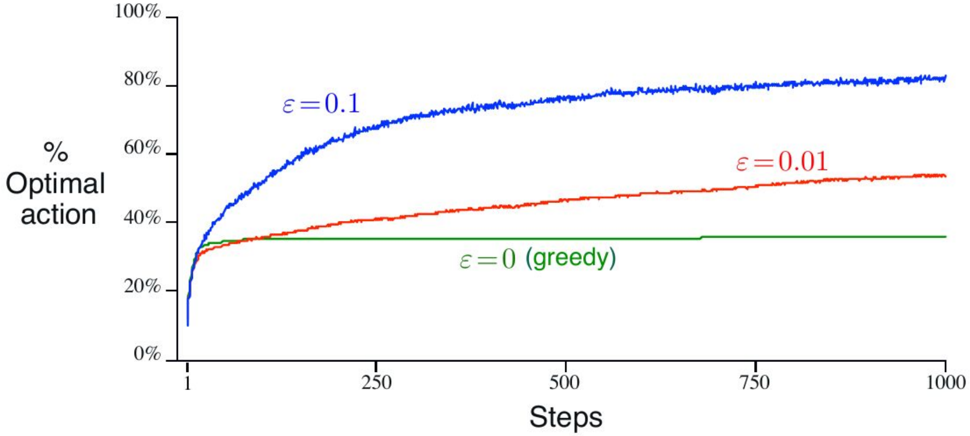


$$R_t \mid A_t = a \sim \mathcal{N}(q^*(a), 1)$$

10-armed Testbed - e-greedy

- Dependiendo del ruído de la recompensa, se modifica el rendimiento del ε greedy.
- Inclusive, en casos donde la recompensa es determinística (en función de la acción) puede convenir ε-greedy (caso no estacionario).





Algoritmo y caso no estacionario

A simple bandit algorithm

Initialize, for $a = 1$ to k:

$Q(a) \leftarrow 0$

$N(a) \leftarrow 0$

Loop forever:

$$A \leftarrow \begin{cases} \arg\max_a Q(a) & \text{with probability } 1-\varepsilon \quad \text{(breaking ties randomly)} \\ \text{a random action} & \text{with probability } \varepsilon \end{cases}$$

$R \leftarrow bandit(A)$

$N(A) \leftarrow N(A) + 1$

$Q(A) \leftarrow Q(A) + \frac{1}{N(A)}\left[R - Q(A)\right]$

Caso no estacionario

$$Q_{n+1} \doteq Q_n + \alpha\Big[R_n - Q_n\Big] = \quad (1-\alpha)^n Q_1 + \sum_{i=1}^{n} \alpha(1-\alpha)^{n-i} R_i.$$

$$\sum_{n=1}^{\infty} \alpha_n(a) = \infty$$

Controla la fluctuaciones del comienzo

$$\sum_{n=1}^{\infty} \alpha_n^2(a) < \infty.$$

Garantiza la convergencia

Método “Optimístico”

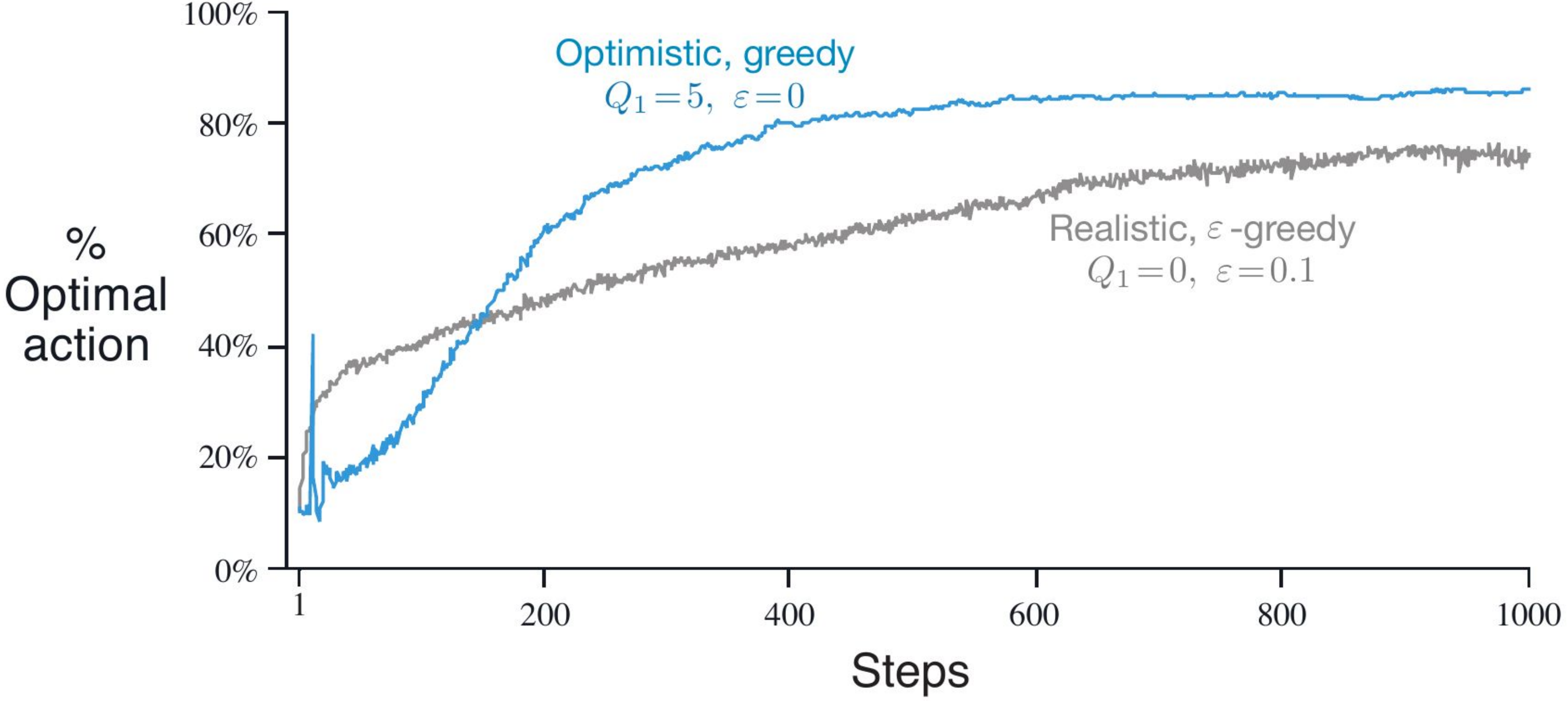
El método anterior es muy dependiente de las *condiciones iniciales*,

$$Q_1(a)$$

$$Q_2(a) = Q_1(a) + \frac{1}{N_1(a)}\left[R_1 - Q_1(a)\right]$$

Si defino todas las condiciones iniciales de manera ***optimista***, la recompensa no va a cumplir con las expectativas por lo que *muchas veces voy a cambiar de acción.*

Es decir, **voy a explorar**!

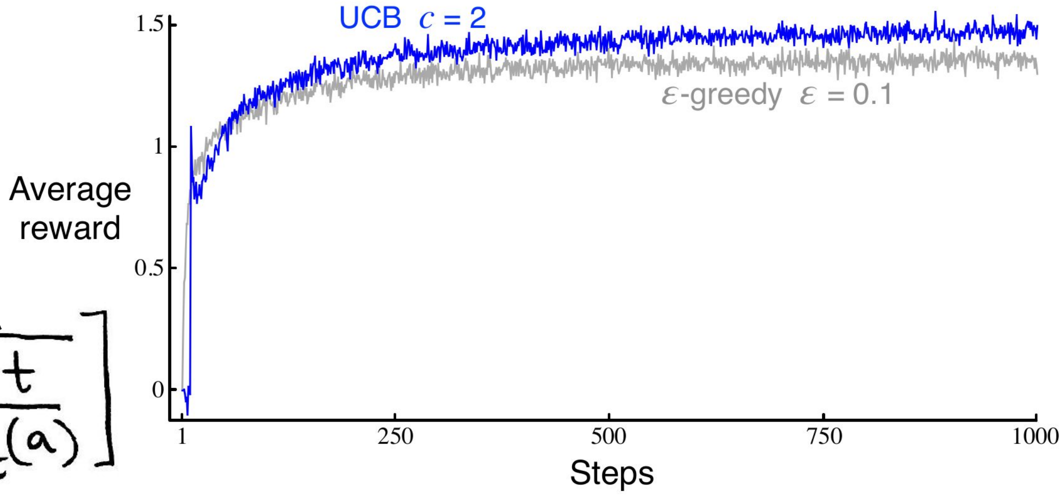

Optimistic, greedy
$Q_1 = 5, \ \varepsilon = 0$
Realistic, ε-greedy
$Q_1 = 0, \ \varepsilon = 0.1$
%
Optimal
action
100%
80%
60%
40%
20%
0%
1
200
400
600
800
1000
Steps

Selección de acción basada en Cota Superior de Confianza (UCB)

e-greedy selecciona las acciones no óptimas *sin utilizar **nada** de información* sobre las mismas.

Cota de la varianza del Q

$$A_t = \underset{a}{\operatorname{argmax}} \left[Q_t(a) + c\sqrt{\frac{\ln t}{N_t(a)}} \right]$$



Programación

Markov Reward Processes

Función de recompensa

$$\mathcal{R}_s := E[R_{t+1} | S_t = s]$$

Retorno

$$G_t := R_{t+1} + \gamma R_{t+2} + \cdots$$

Función de Valor

$$v(s) = E[G_t | S_t = s]$$

Ecuación de Bellman

$$v(s) = \mathcal{R}_s + \gamma \sum_{s'} p_{s,s'} v(s')$$

Un sistema de K ecuaciones!

$$v = (v(s_1), \ldots, v(s_K))$$

$$v = \mathcal{R} + \gamma p v$$

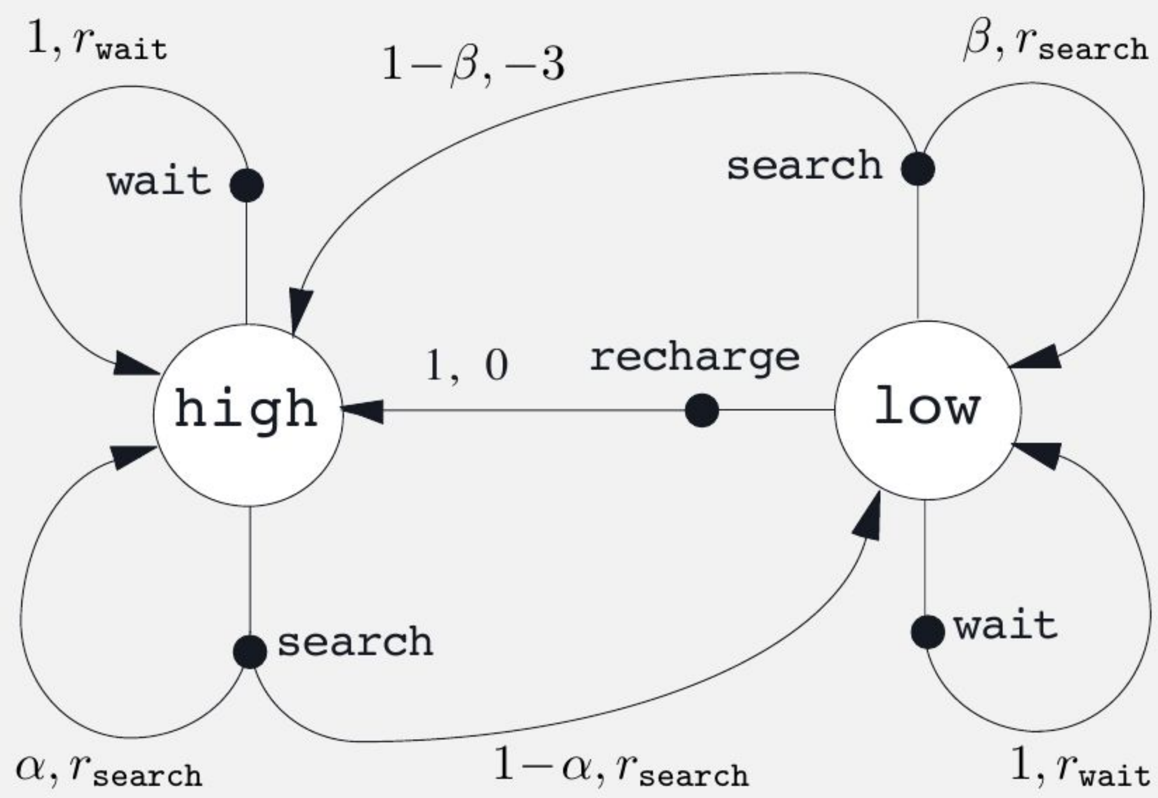
Proceso Markovianos de Decisión

Se agrega un *acción* la cual modifica el ambiente.

$$p^{a}_{s,s'} := P(S_{t+1} = s' \,|\, S_t = s, A_t = a)$$

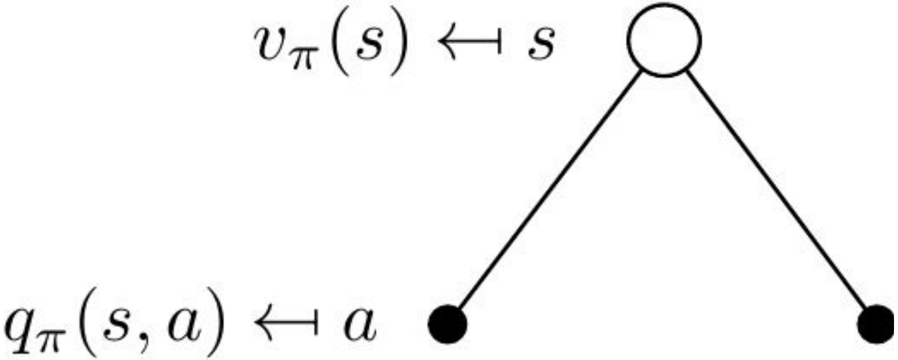
$$\mathcal{R}^{a}_{s} := E[R_{t+1} \,|\, S_t = s, A_t = a]$$

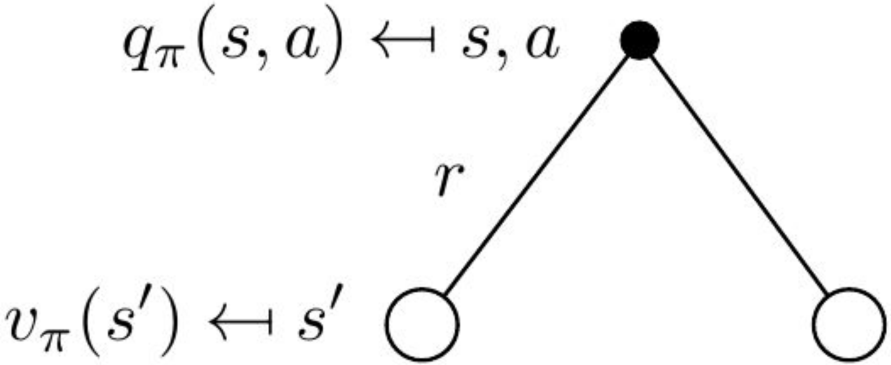
Un ejemplo



$1, r_{\text{wait}}$
$1-\beta, -3$
β, r_{search}
wait
search
high
$1, \ 0$
recharge
low
search
wait
$\alpha, r_{\text{search}}$
$1-\alpha, r_{\text{search}}$
$1, r_{\text{wait}}$

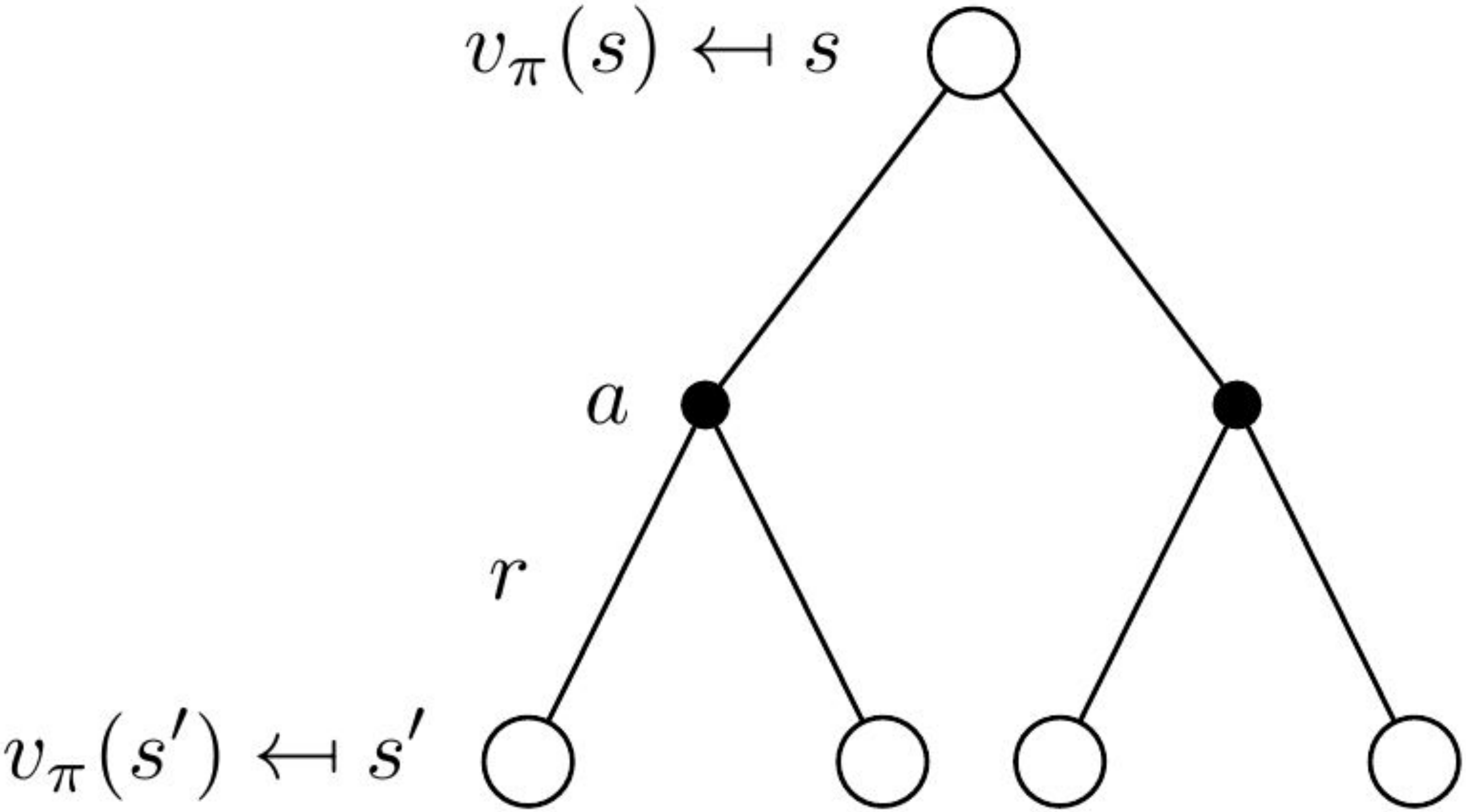
Ecuación de Bellman (bis)





$$v_\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s) q_\pi(s,a)$$

$$q_\pi(s,a) = \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_\pi(s')$$


$v_\pi(s) \leftarrow s$
a
r
$v_\pi(s') \leftarrow s'$

Evaluación de Política

$$v_\pi(s) = \sum_a [\mathcal{R}_s^a + \sum_{s'} v_\pi(s') p_{s,s'}^a] \pi(a|s)$$

$$= \mathcal{R}_s^\pi + \sum_{s'} v_\pi(s') p_{s,s'}^\pi$$

Método Iterativo

$$v_\pi^{k+1}(s) = \mathcal{R}_s^\pi + \sum_{s'} v_\pi^{k}(s') p_{s,s'}^\pi$$

Función de Valor Óptima

$$v_*(s) = \max_{\pi} v_\pi(s)$$

$$q_*(s, a) = \max_{\pi} q_\pi(s, a)$$

Optimalidad de MDP

$$\exists\ \pi_* \ / \pi_* \geq \pi\ \forall\ \pi \qquad \text{such that}$$

$$v_*(s) = v_{\pi_*}(s), \qquad q_*(s,a) = q_{\pi_*}(s,a) \qquad \forall\ s, a$$

$$\pi_*(s) = arg\,\mathrm{max}_a\, q_*(s,a)$$

$$v_*(s) = \mathrm{max}_a\, q_*(s,a)$$

Optimalidad de Bellman

$$T_{ij} = \text{Costo de viajar de } i \text{ a } j$$

$$O_{ij} = \text{Costo del viaje ÓPTIMO de } i \text{ a } j$$

$$O_{ij} = min_k [T_{ik} + O_{kj}]$$

Ecuaciones de optimalidad para MDP

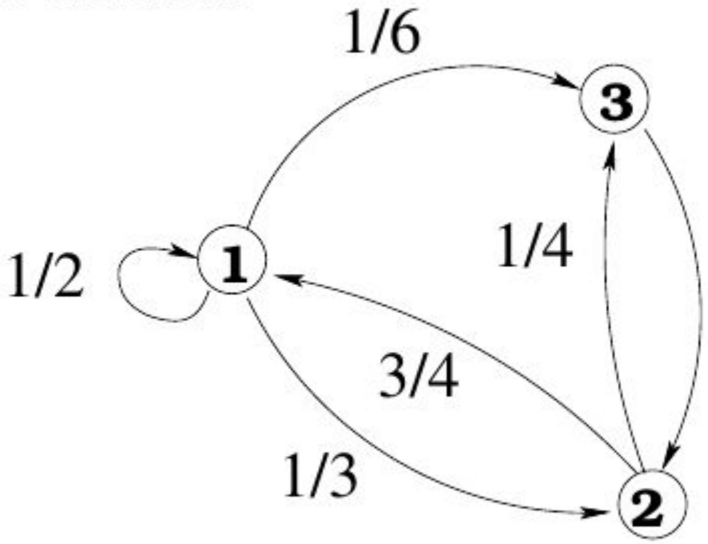
$$v_*(s) = \max_a \left[\mathcal{R}_s^a + \sum_{s'} p_{s,s'}^a v_*(s')\right]$$

Son ecuaciones NO lineales!

¿Cómo obtener v_* y π_*?

Programación

Ejercicio para entregar



$$R(1) = -2, \qquad R(2) = 3, \qquad R(3) = 5$$

$$G(S_1, S_2) = R(S_1) + R(S_2)$$

Calcular $E_{\mathbf{e1}}[G(S_1, S_2)]$ de manera analítica y por monte carlo.

Lectura recomendada:

- Introducción a Contextual Bandits: https://towardsdatascience.com/contextual-bandits-and-reinforcement-learning-6bdfeaece72a
- Capítulos 2 y 3 de Sutton and Barto
- Desafíos de Reinforcement Learning: https://www.alexirpan.com/2018/02/14/rl-hard.html