Aprendizaje Reforzado
Clase 2

Julián Martínez

Dada una política, ¿Cómo calcular la función de valor?

$$v(s) = E[G_t \mid S_t = s] = E[R_{t+1} + \gamma G_{t+1} \mid S_t = s]$$

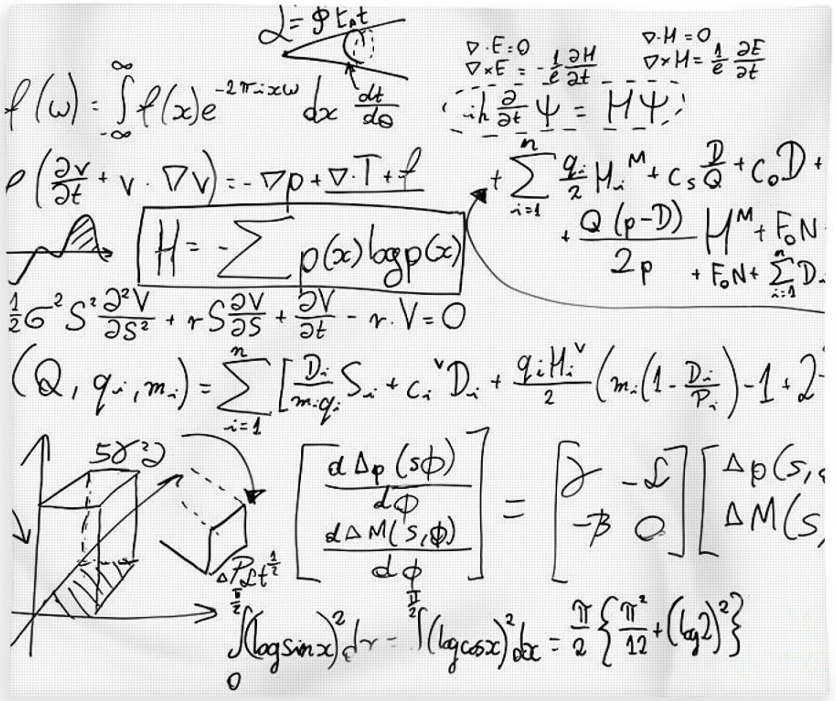
$$= R(s) + \gamma E[G_{t+1} \mid S_t = s]$$

$$= R(s) + \gamma \sum_{s'} E[G_{t+1} \mid S_{t+1} = s'] P(S_{t+1} = s' \mid S_t = s)$$

$$= R(s) + \gamma \sum_{s'} v(s') P^{\pi}_{ss'}$$

Ecuación de Bellman

Sobre la ecuación de Bellman



- Sistema de ecuaciones lineales
- Mezcla de cadenas de markov

Algoritmo Iterativo (Evaluation)(Jacobi, Gauss-Seidel)

$$v^{k+1}(s) = R(s) + \gamma \sum_{s'} v^{k}(s') \, P^{\pi}_{ss'}$$

O puedo ir reemplazando a medida que calculo

$$v^{k}(s) \leftrightarrow v^{k+1}(s)$$

¿Por qué converge?

Bellman
Expectation
Backup Operator

$$T^{\pi}(v) := R^{\pi} + \gamma P^{\pi} v$$

$$\| T^{\pi}(u) - T^{\pi}(v) \|_{\infty} = \| \gamma P^{\pi}(u - v) \|_{\infty}$$

$$\leq \gamma \| u - v \|_{\infty}$$

Contraction Mapping Theorem

Política óptima

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

Función de estado-Acción

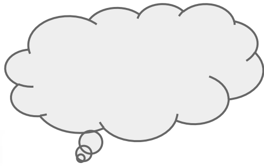
$$q_{\pi}(s,a) := \mathbb{E}_{\pi}\left[G_t \mid S_t = s, A_t = a\right]$$

¿Cómo mejorar una política?

$$v_\pi(s) = E\left[G_t \mid S_t = s\right]$$

$$= \sum E\left[G_t \mid S_t = s, A_t = a\right] \times P\left(A_t = a \mid S_t = s\right)$$

$$v_\pi(s) = \sum_a q_\pi(s,a)\, \pi(a \mid s)$$



Algunas observaciones

$$v_\pi(s) = \sum_a q_\pi(s,a)\,\pi(a|s)$$

$$\pi^+(s) := \underset{a}{\operatorname{argmax}}\; q_\pi(s,a)$$

$$v_\pi(s) \leq q_\pi(s, \pi^+(s))$$

Efectivamente, mejora la función de valor

$$v_\pi(s) \leq q_\pi(s, \pi'(s)) = E\left[R_{t+1} + \gamma v_\pi(S_{t+1}) \,\middle|\, S_t = s, A_t = \pi'(s)\right]$$

$$= E_{\pi'}\left[R_{t+1} + \gamma v_\pi(S_{t+1}) \,\middle|\, S_t = s\right]$$

Como π' determinística $\Rightarrow A_t = \pi'(S_t)$

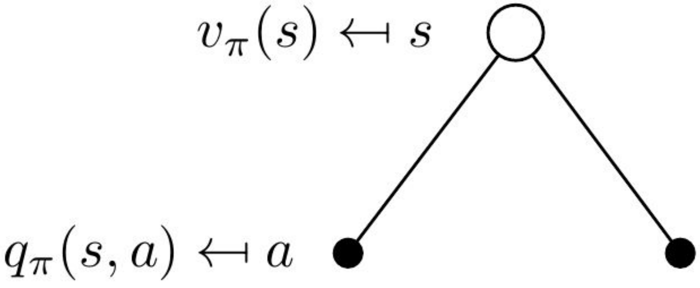
$$\leq E_{\pi'}\left[R_{t+1} + \gamma q_\pi(S_{t+1}, \pi'(S_{t+1})) \,\middle|\, S_t = s\right]$$

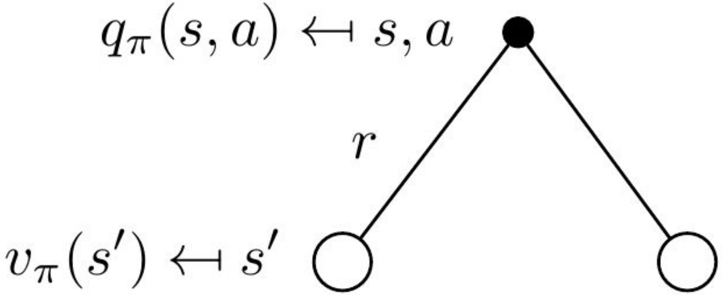
Condiciono en A_t

(V-EA) $$v_\pi(s) = \sum_a q_\pi(s,a)\, \pi(a|s)$$

Condiciono en S_{t+1}

(EA-V) $$q_\pi(s,a) = \sum_{s'} \left\{ r(s,a,s') + \gamma\, v_\pi(s') \right\} \cdot p^{a}_{s,s'}$$





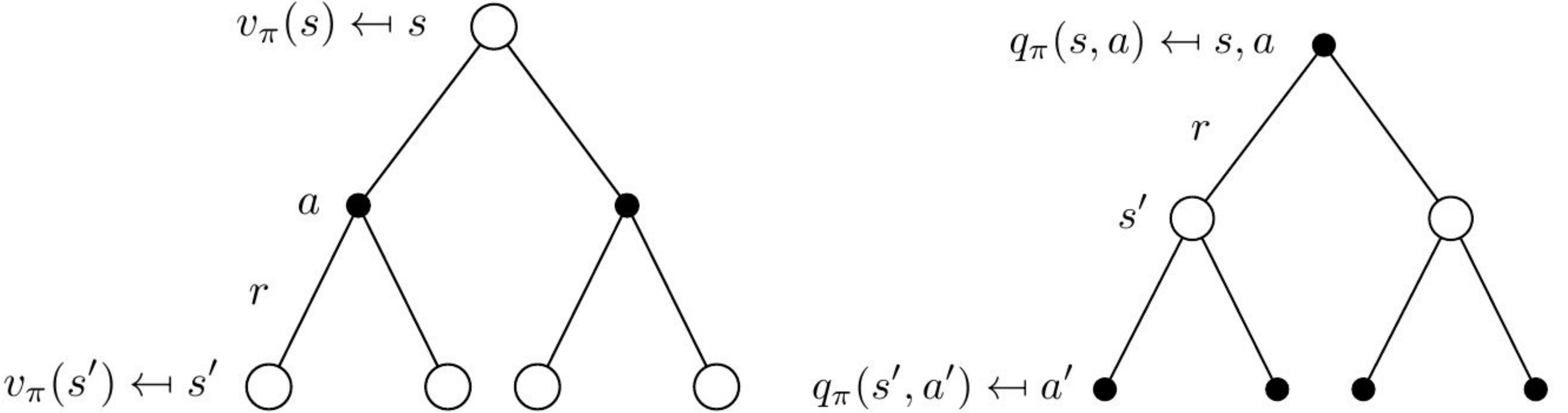
Existencia de Política Óptima

Theorem

For any Markov Decision Process

- *There exists an optimal policy* π_* *that is better than or equal to all other policies,* $\pi_* \geq \pi, \forall \pi$
- *All optimal policies achieve the optimal value function,* $v_{\pi_*}(s) = v_*(s)$
- *All optimal policies achieve the optimal action-value function,* $q_{\pi_*}(s,a) = q_*(s,a)$

Tomado de los slides de David Silver


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

Ecuaciones de Optimalidad de Bellman

$$v_*(s) = \max_{a \in A(s)} q_{\pi_*}(s,a)$$

$$v_\pi(s) = \sum_a q_\pi(s,a)\, \pi(a|s)$$

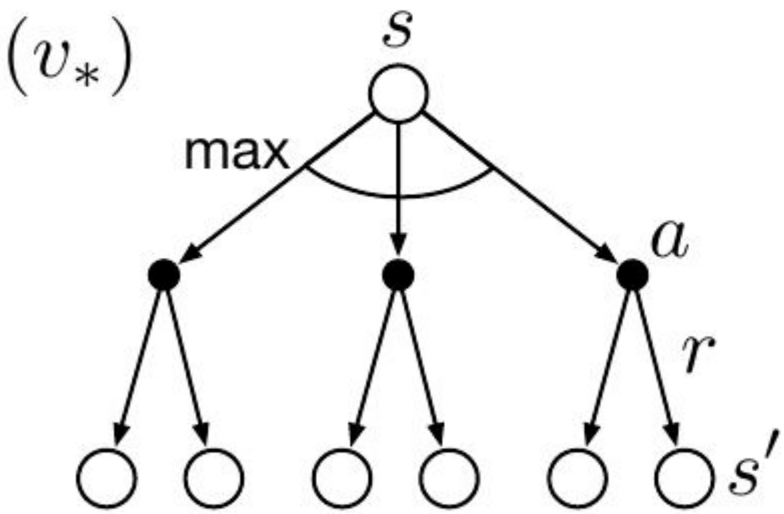
$$q_\pi(s,a) = \sum_{s'} \left\{ r(s,a,s') + \gamma\, v_\pi(s') \right\} \cdot p^a_{s,s'}$$

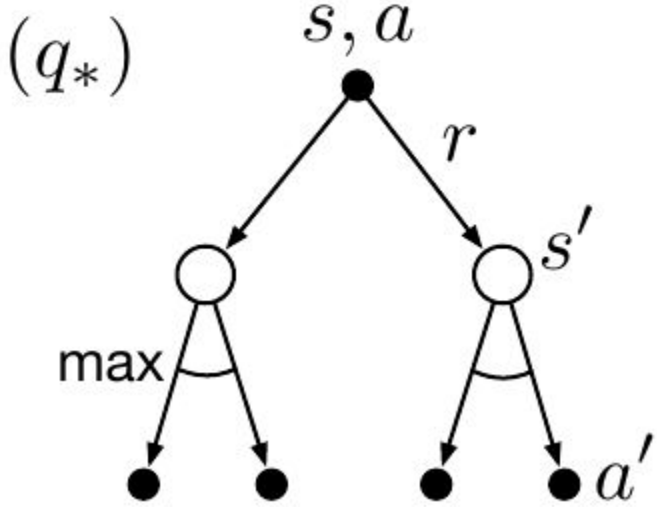
$$v_*(s) = \max_a \left[\sum_{s'} \left\{ r(s,a,s') + \gamma\, v_*(s') \right\} p^a_{s,s'} \right]$$

$$q_*(s,a) = \sum_{s'} \left\{ r(s,a,s') + \gamma \max_{a'} q_*(s',a') \right\} p^a_{s,s'}$$

- Algoritmos iterativos
- Ecuaciones NO lineales!

Backup diagrams





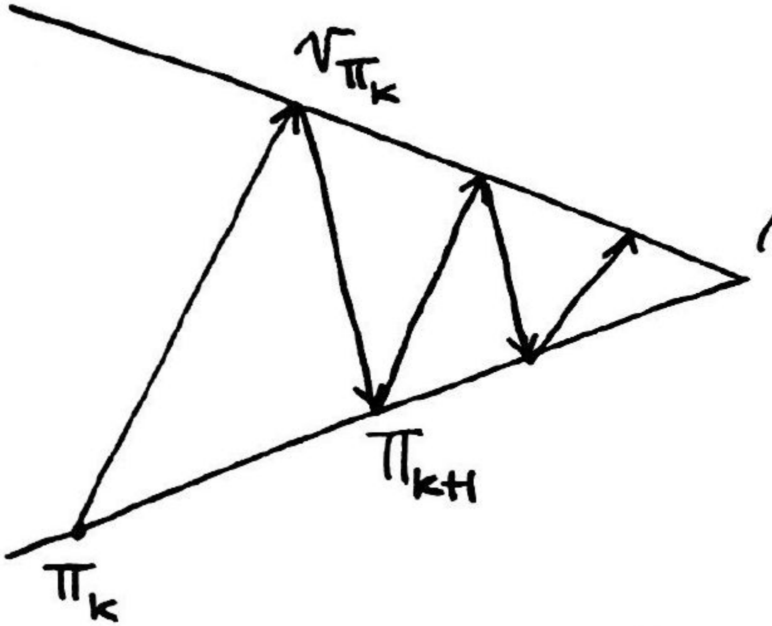
Optimalidad de Bellman

T_{ij} = Costo de viajar de i a j

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

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

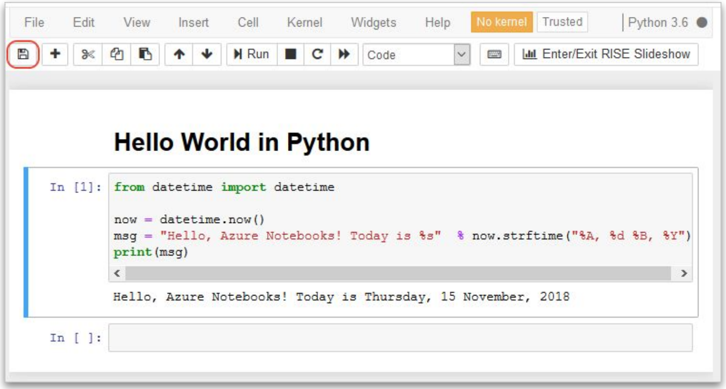
Evaluación y mejora



$$\pi_{k+1}(s) := \arg\max_a q_{\pi_k}(s,a)$$

$$\pi_k \longrightarrow v_{\pi_k} \longrightarrow q_{\pi_k} \longrightarrow \pi_{k+1}$$

Introducción a OpenAI GYM



- Introducción General
- Para que “jueguen”: Mountain
- Ejemplo del Robot (Batería)

Introducción a OpenAI Gym

- ¿Cómo instalarlo? ubuntu 18.04, Python, jupyter, open ai gym
  - Linux:
    - sudo apt-get update
    - sudo apt-get install python3 python3-pip ipython3 python3-fontconfig
    - sudo apt-get install libglu1-mesa-dev freeglut3-dev mesa-common-dev python-opengl
    - pip3 install numpy pandas matplotlib jupyter gym
  - Windows:
    - Virtual Box:
      - instalar ubuntu 16.04 y usar el instructivo de la parte de Linux
    - WSL:(inspirado en https://github.com/openai/gym/issues/11#issuecomment-242950165)
      - instalar Windows Subsystem for Linux (WSL): https://docs.microsoft.com/en-us/windows/wsl/install-win10
      - instalar ubuntu 16.04 LTS para WSL yendo a Microsoft Store (barra de búsqueda de Windows) y buscando Ubuntu 16.04
      - correr una consola WSL (buscar Ubuntu en la barra de búsqueda de Windows)
      - realizar los mismos pasos que en el instructivo de linux en esa consola
      - instalar vcXsrv/xming;
      - correr vcXsrv (elegir one large window); tipear en la consola de comandos de WSL: export DISPLAY=:0
  - Correr jupyter: jupyter notebook --no-browser
  - Google colab: ir a google colab, https://colab.research.google.com/notebooks/welcome.ipynb#recent=true, elegir la solapa Github y buscar en https://github.com/javkrei/aprendizaje-reforzado-austral