APRENDIZAJE POR REFUERZO

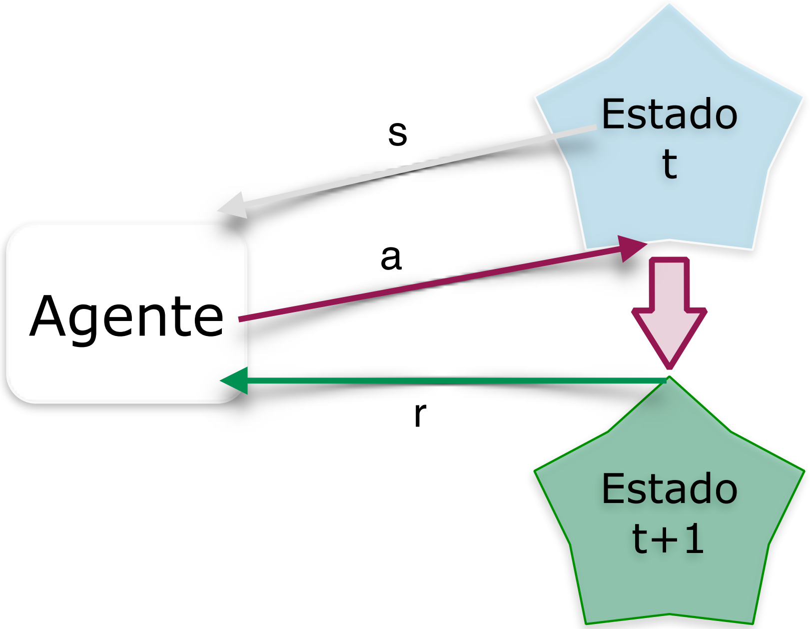
Índice

- Definición
- Aprendizaje Q
- Ambiente no-determinista
- Diferencia Temporal
- Representación / Generalización

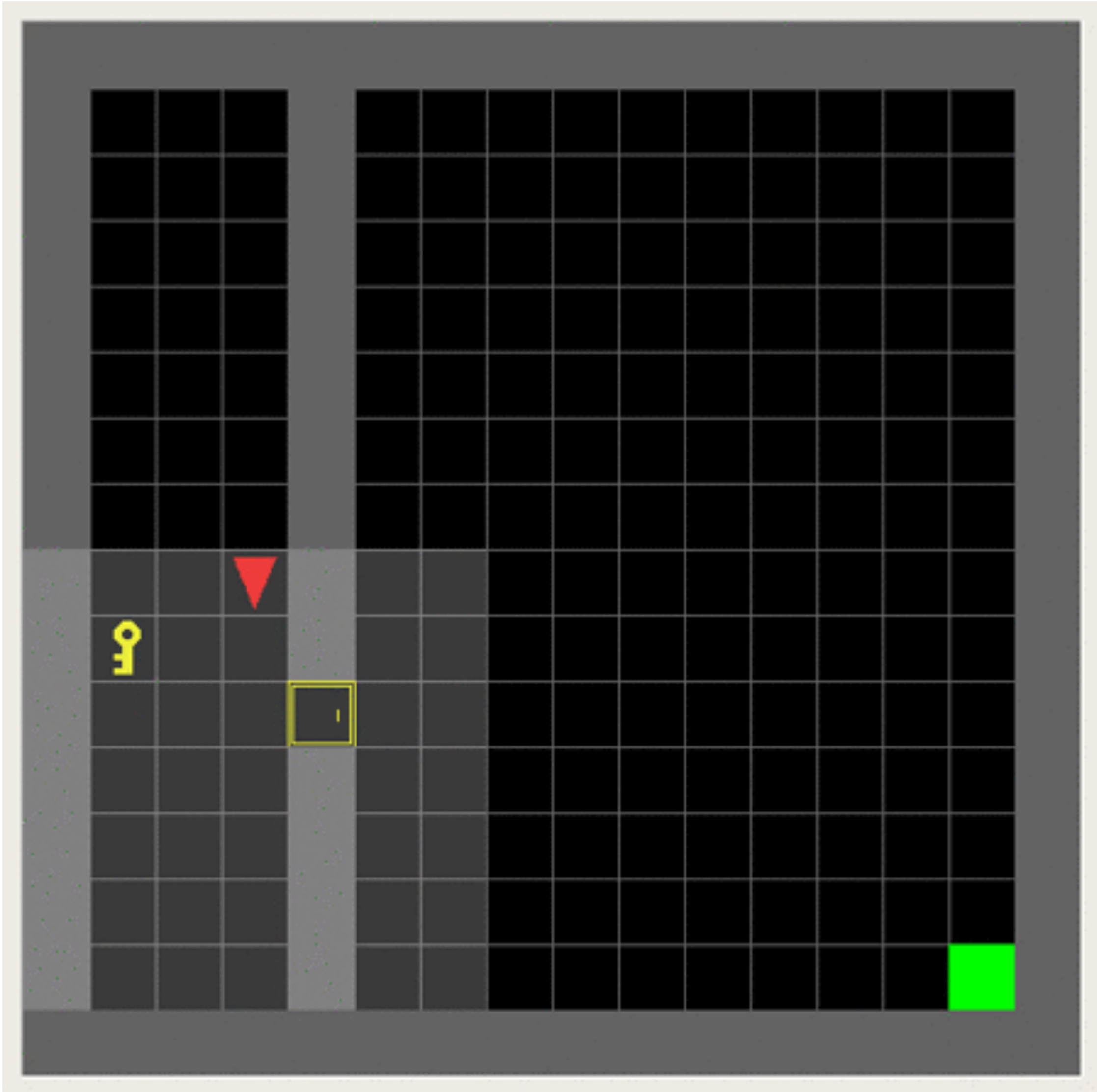
Definición

- La tarea consiste en aprender una estrategia de comportamiento para un agente de forma de realizar un objetivo dado

- Se considera que:
  - El agente se encuentra en un estado **s**, que puede sensar de alguna forma.
  - El agente puede elegir realizar una acción **a** en ese estado.
  - Su acción provoca un cambio en el mundo (un nuevo estado).
  - El comportamiento es calificado con una recompensa **r**.



Definición



Definición

- Características de este tipo de aprendizaje:

  - No contamos con un conjunto de instancias $<s, \pi(s)>$
  - No siempre se tiene una “recompensa” inmediata a una acción.
  - No siempre una misma acción conduce a un mismo estado.
  - Observación parcial de los estados.
  - La política se aprende a medida que se la ejecuta.

Definición

- Entonces:
  - ¿Cómo se distribuye la recompensa en toda la cadena de acciones?
  - Exploración vs. explotación: ¿cómo se balancea la búsqueda de nuevos datos con la obtención recompensas a partir de lo ya aprendido?
  - Aprovechamiento de la información ya recolectada.

Definición

- Escenarios posibles:
  - ¿Son las acciones del agente deterministas?
  - ¿Puede el agente predecir el resultado de su acción?
  - ¿Se cuenta con un experto que enseñe?
  - ¿Se puede elegir la secuencia de entrenamiento?
- Consideremos procesos de decisión de Markov (MDPs):
  - El tiempo es discreto.
  - El agente selecciona una acción a_t en estado s_t
  - El ambiente retorna una recompensa $r_t=r(s_t,a_t)$ y el siguiente estado $s_{t+1}=\delta(s_t,a_t)$. El agente desconoce estas funciones.
  - Nada depende de la secuencia de acciones previas al estado s_t.

Definición

- Buscamos una secuencia de acciones π que maximice el retorno acumulado con descuento:

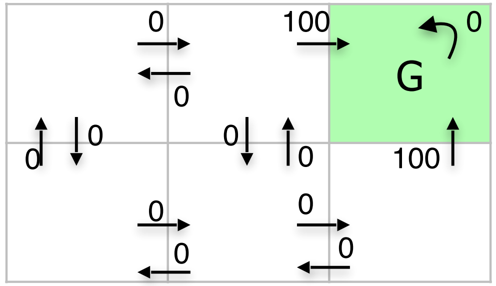
$$V_\pi(s_t) = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots = \sum_i \gamma^i r_{t+i}$$

donde 0≤γ<1 pondera el retorno inmediato vs. el futuro

- La estrategia óptima π* será aquella que maximiza el retorno:

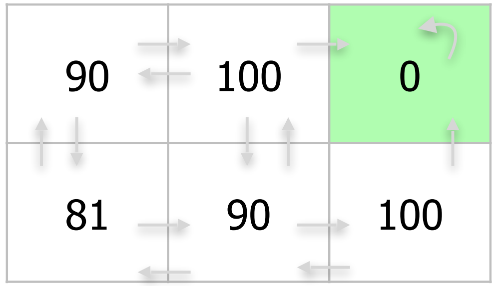
$$\pi^*(s) = argmax_\pi V_\pi(s), \forall s \qquad V^*(s) = V_{\pi^*}(s)$$

- Otras posibles V:
  - horizonte finito: $\sum_i^h \gamma^i r_{t+i}$
  - recompensa media: $lim_{h \to \infty} \frac{1}{h} \sum_i^h \gamma^i r_{t+i}$

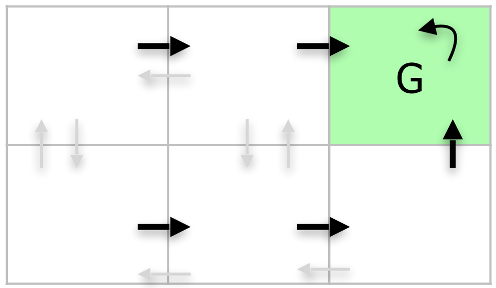


r(s,a)

con γ=0,9 :



V*(s)



π*(s)

Aprendizaje Q

- ¿Cómo obtenemos π^*?
  - Intentamos aprender V^*.
  - Luego, la acción a tomar es la que lleva al siguiente estado que maximiza la recompensa:

$$\pi^*(s) = argmax_{a \in A}[r(s,a) + \gamma V^*(\delta(s,a))]$$

    - Se requiere conocer r y δ.
    - ¿Qué sucede cuando no se cuenta con información perfecta?

Aprendizaje Q

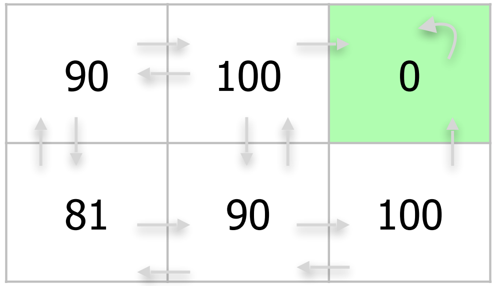
- Utilizamos otra función de evaluación:

$$Q(s,a) = r(s,a) + \gamma V^*(\delta(s,a))$$

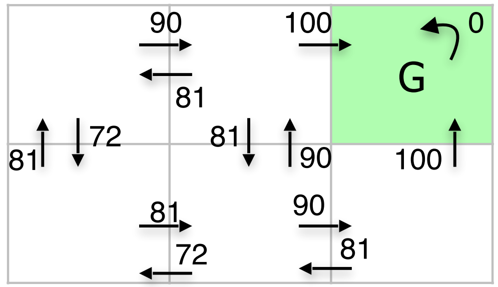
- La acción a tomar es :

$$\pi^*(s) = argmax_{a \in A} Q(s,a)$$

- Esto es simplemente una reescritura, pero si logro aproximar Q, no preciso conocer directamente a r ni a δ.
- Para aproximarla tomamos en cuenta la definición de V*:



V*(s)



Q(s,a)

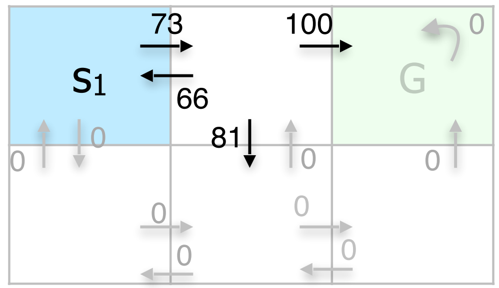
$$\left.\begin{aligned} Q(s,a) &= r(s,a) + \gamma V^*(\delta(s,a)) \\ V^*(s) &= max_a Q(s,a) \end{aligned}\right\} Q(s,a) = r(s,a) + \gamma max_{a_2} Q(\delta(s,a), a_2)$$

Aprendizaje Q

- Estimamos la función Q:

$$\hat{Q}(s,a) \leftarrow r(s,a) + \gamma max_{a_2} \hat{Q}(\delta(s,a), a_2)$$

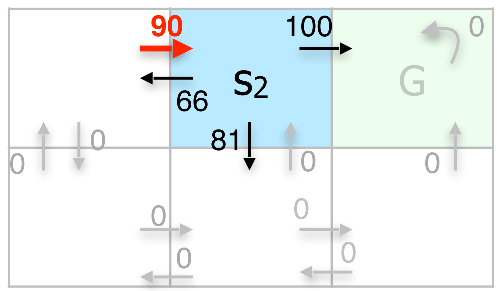
  - Inicializo la tabla con valores nulos.
  - Elijo una acción.
  - Veo el resultado: r y s'.
  - Actualizo Q(s,a).
- Observar que en el ejemplo:
  - En la primer corrida no se actualiza ninguna entrada hasta llegar a G.
  - En cada corrida los valores se propagan hacia atrás a medida que $\hat{Q}$ se actualiza.



derecha $\hat{Q}(s,a)$ en t

$\hat{Q}(s_1, derecha) \leftarrow$
0 + 0,9 max({66,81,100})

$\hat{Q}(s_1, derecha) \leftarrow$ 90



$\hat{Q}(s,a)$ en t+1

Aprendizaje Q

- Convergencia:

  Sea un MDP determinista con recompensa acotada. La tabla de $\hat{Q}$ se inicializa con valores aleatorios. Si el agente visita todo estado-acción infinitas veces, $\hat{Q}$ converge a Q.

Explotación vs. Exploración

- Lo razonable sería elegir siempre la de mayor recompensa estimada. (explotación)
- Pero se pueden perder otras mejores, además de no cumplir la visita "infinita" a todos los pares (s,a). (exploración)
- Se puede establecer una política de exploración aleatoria, por ejemplo, ponderada por lo ya conocido...

$$P(a_i \mid s) = \frac{k^{\hat{Q}(s,a_i)}}{\sum_j k^{\hat{Q}(s,a_j)}}$$

¿Cómo agilitar el aprendizaje?

- Podemos repetir un mismo episodio (en memoria) las veces que queramos...
- Actualizar la secuencia en orden inverso a su ejecución.
- Recolectar los <s,a,r> y reentrenar periódicamente con estos valores.

Aprendizaje TD

- El algoritmo Q es un caso particular de aprendizaje por diferencia temporal.
- Estos algoritmos reducen la diferencia de estimación que hace un agente con el paso del tiempo:

$$Q^{(1)}(s_t, a_t) \leftarrow r_t + \gamma max_a \hat{Q}(s_{t+1}, a)$$ miro 1 paso delante

$$Q^{(2)}(s_t, a_t) \leftarrow r_t + \gamma r_{t+1} + \gamma^2 max_a \hat{Q}(s_{t+2}, a)$$ miro 2 pasos

...

$$Q^{(n)}(s_t, a_t) \leftarrow r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots + \gamma^{n-1} r_{t+n-1} + \gamma^n max_a \hat{Q}(s_{t+n}, a)$$ miro n pasos

- Todas esas fórmulas a su vez se pueden combinar en una:

$$Q^{\lambda}(s_t, a_t) = (1 - \lambda) \,.\, [Q^{(1)}(s_t, a_t) + \lambda Q^{(2)}(s_t, a_t) + \lambda^2 Q^{(3)}(s_t, a_t) + \ldots]$$

- Cuanto mayor es el valor de λ, más pesan los valores más alejados.

$$Q^{\lambda}(s_t, a_t) \leftarrow r_t + \gamma[(1 - \lambda) max_a \hat{Q}(s_{t+1}, a) + \lambda Q^{\lambda}(s_{t+1}, a_{t+1})]$$

Ambiente no-determinista

- ¿Qué sucede cuando acciones y recompensas no son deterministas?
- Generalizamos el algoritmo de aprendizaje Q:

$$V_\pi(s_t) = E(\sum_i \gamma^i r_{t+i})$$

$$Q(s,a) = E(r(s,a) + \gamma V^*(\delta(s,a))) = E(r(s,a)) + \gamma E(V^*(\delta(s,a)))$$

$$= E(r(s,a)) + \gamma \sum_{s'} P(s' | s,a) V^*(s')$$

$$= E(r(s,a)) + \gamma \sum_{s'} P(s' | s,a) \,.\, max_{a'} Q(s',a')$$

- Como r no es determinista, la anterior regla de aprendizaje no converge; utilizamos entonces la siguiente regla:

$$\hat{Q}(s,a) \leftarrow (1-\alpha_n) \,.\, \hat{Q}(s,a) + \alpha_n [r(s,a) + \gamma max_{a_2} \hat{Q}(\delta(s,a), a_2)]$$

donde α_n actúa como tasa de aprendizaje, por ejemplo:

$$\alpha_n = \frac{1}{visitas(s,a)}$$

Ambiente no-determinista

- Convergencia:

(H) Sea un MDP no determinista con recompensa acotada, y n(i,s,a) la iteración correspondiente a la i-ésima vez que la acción a se aplica en s.

Si todo par estado-acción se visita infinitas veces, $0 \leq \alpha_n < 1$ y:

$$\sum_{i=1}^{\infty} \alpha_n(i,s,a) = \infty \qquad \sum_{i=1}^{\infty} \alpha_n(i,s,a)^2 < \infty$$

(T) $\hat{Q}$ converge a Q

Representación de Q

- No siempre se puede tener una tabla para representar Q.
- Además, se puede intentar generalizar Q a partir de los ejemplos vistos.
- La función, entonces, se puede representar con una función lineal, una red neuronal, etc.
- Problema: la convergencia de $\hat{Q}$ a Q no está garantizada.

Representación de Q

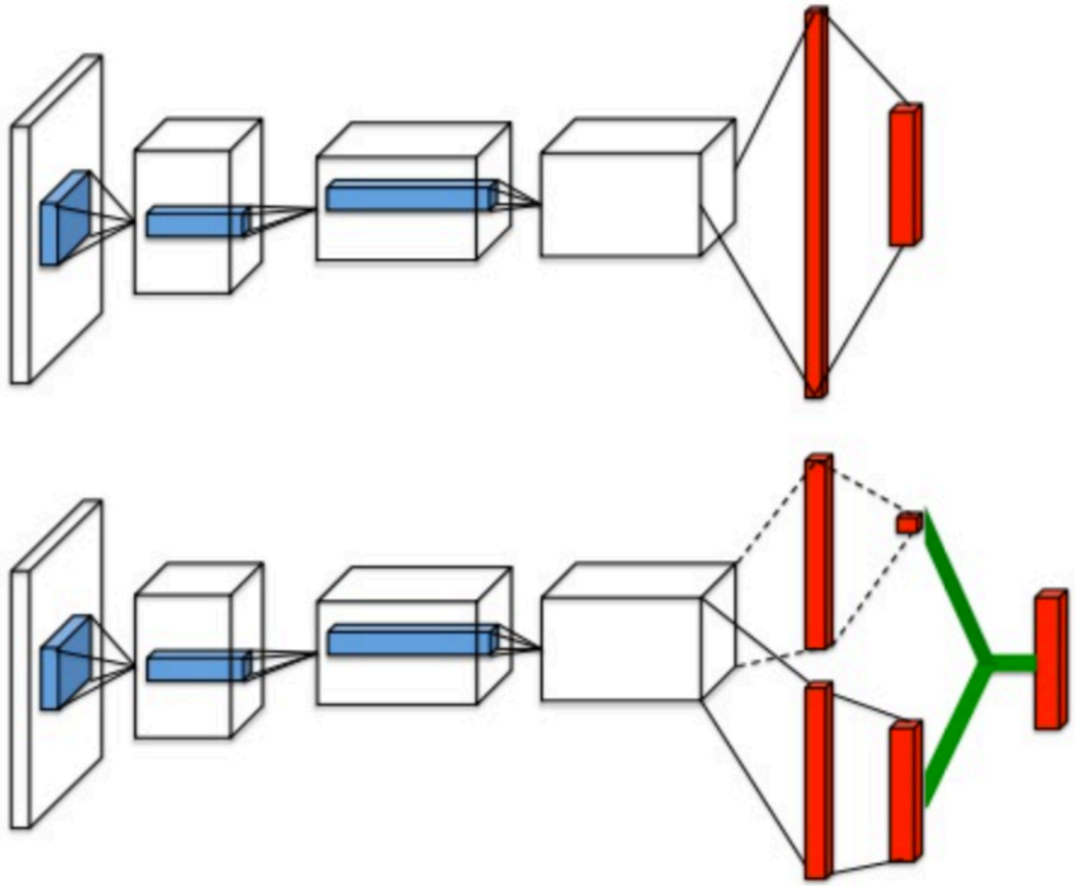


Figura 2.12: Arriba: Q-network estándar [29] con una única secuencia. Abajo: *Dueling Q-network*. Esta red tiene dos secuencias para estimar separadamente el escalar $V(s)$ y la ventaja para cada acción. Tomada del trabajo de van Hasselt [43].

Representación de Q

Refuerzos y una vida de juegos :)

- 1963 - Ta-Te-Ti (Menace)
- 1992 - Backgammon (TD-Backgammon)
- 2016-2017 - Go (AlphaGo - AlphaGo Zero)
- 2018 - Ajedrez, Shogi y Go (AlphaZero)
- 2019 - Atari* (MuZero)