Breve introducción a Reinforcement Learning

A. Atutxa

LSI BIlbao

Diciembre 2021

[1] Basado en el libro de Sutton y Barto, curso de Adam y Martha White (U. Alberta), curso de UCL D. Silver, curso de Standford y cursos de de E. BrunSkill, Thomas Simonini, DeepMind y

Overview

Toma de decisiones secuencial

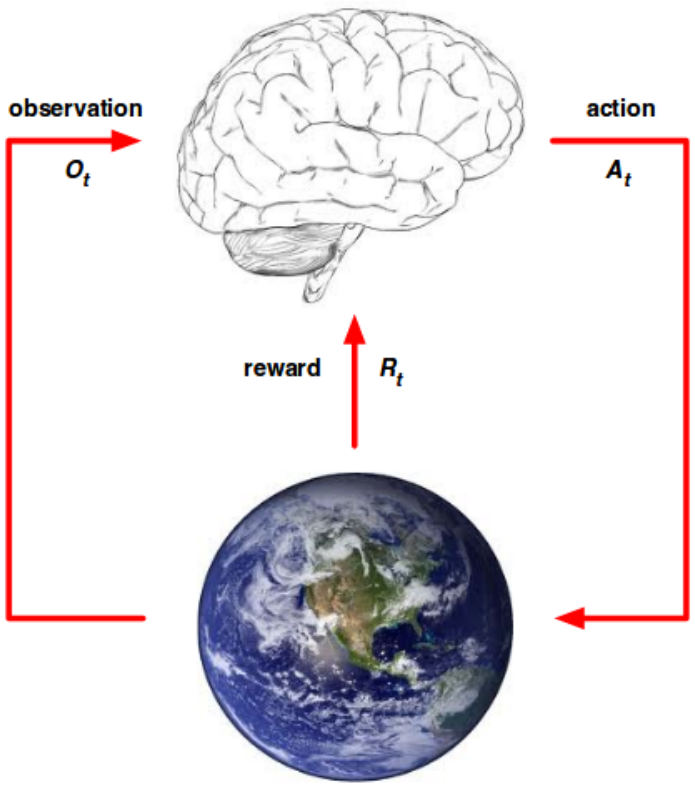
- **Objetivo de RL:** Seleccionar las acciones que maximicen el futuro premio acumulado
  - Cada acción puede tener consecuencias a largo plazo.
  - El premio no tiene porque ser inmediato
  - El mejor premio a corto plazo no tiene por qué ser el mejor a largo plazo. Actuar de forma Greedy no siempre es la mejor estrategia (p.e. las inversiones)

RL se basa en la siguiente hipótesis (premisa):

Definición (La hipotesis del premio)

Todo Goal puede ser descrito como una maximización del cúmulo de premios esperado

Contexto: Agente y Entorno



- En cada paso t el agente:
  - Recibe/Percibe una observación
  - Ejecuta una acción
  - Recibe un premio
- En cada paso t el entorno:
  - Emite una observación
  - Recibe una acción
  - Emite un premio

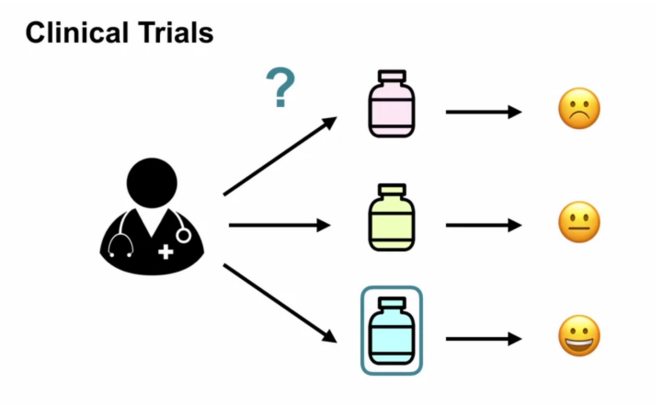
El problema de los k-armed bandits (las K máquinas tragaperras)

- Ejemplo básico de la *biblia* de RL (libro de Richard S. Sutton y Andrew G. Barto[1])
- Nos va a permitir:
  - Formalizar **la toma de decisiones** bajo **incertidunbre**
  - Entender: **acción, premio, valor de una acción**
- Ejecutar :
  `https://mdp.ai/coursera/c01-k-armed-bandit/`

[1] `https://web.stanford.edu/class/psych209/Readings/SuttonBartoIPRLBook2ndEd.pdf`

K-bandits vs. Markov Decision Processes

Consecuencias de cada accion en el entorno: No influencia sobre posteriores premios



Consecuencias de cada accion en el entorno: influencia sobre posteriores premios

Formalización

Formalización como proceso de decisión de Markov

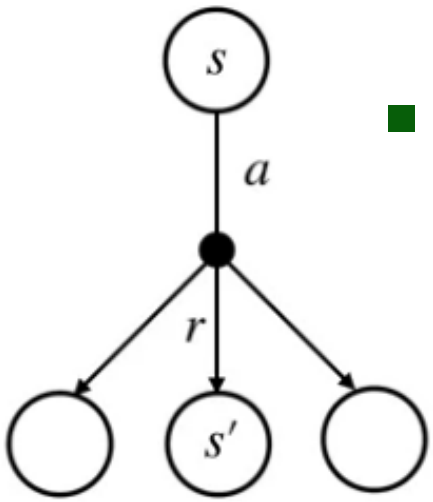
- $M =< S, >, T, R >$
- S: Conjunto finito de estados, $S_t \in S$
- A: Conjunto finito de acciones disponibles. $A_t \in A(S_t)$. A_t es la acción en el instante t que pertenece a las acciones disponibles en el estado S_t.
- T: Función de transición. Cuando se trata de un entorno estocástico $T : S \times A \times S \rightarrow P(S)$.

$$T(s'|s,a) = Pr(S_{t+1} = s'|S_t = s, A_t = a)$$

$$\sum_{s' \in S} T(s'|s,a) = 1$$

- $R : S \times A \times S \wedge R \in \mathbb{R}$

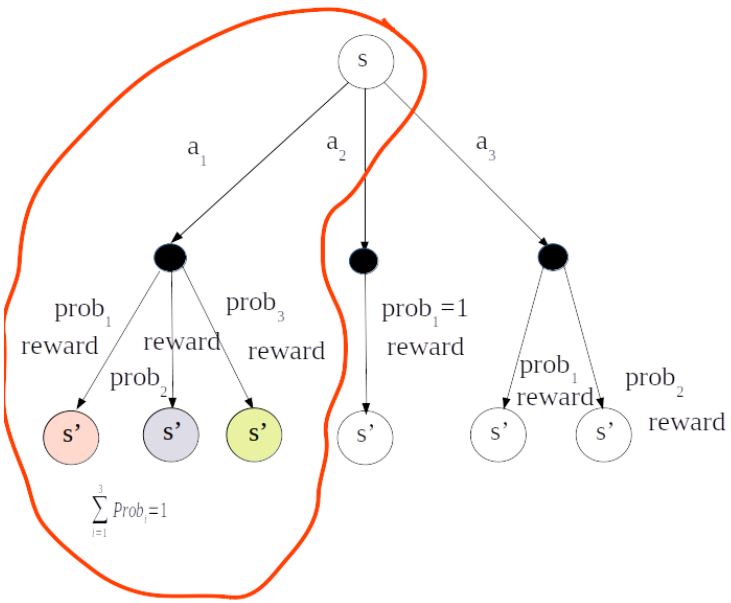
Contexto: Agente y Entorno en MDPs



- En s realizamos a
- Recibiremos un premio (r) y alcanzaremos el estado s´, dependiendo de la distribución de probabilidad oculta
- En el ejemplo del pacman si nos decidimos a ir a la izquierda:
  - con prob(X): premio +10 y $s´ =$ la cereza no está y el fantasma se ha movido hacia la izda
  - con prob(1-X): premio -100 y $s´ =$ la cereza y el pacman no están!!, porque el fantasma se ha movido hacia la derecha y nos ha comido!!

MDPs

- **Modelo de Transiciones:** función de transición T, es decir, la probabilidad de trasicionar el estado actual S realizando una acción A a los siguiente estados s´ posibles.

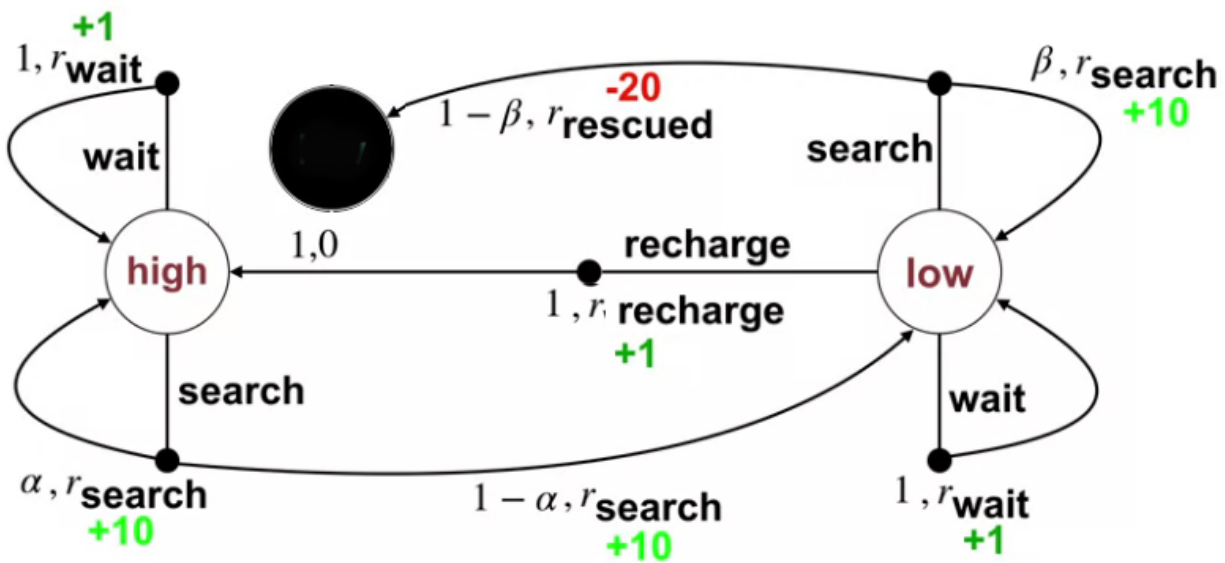


- **Modelo de Premios:** función de los premios R, es decir, el valor del premio que podríamos obtener dado un estado y una acción al pasar a los distintos estados s´ posibles.

Modelar un MDP

- **Objetivo del robot:** Recoger el máximo de latas hasta gastarse la batería

Modelar un MDPs



Política

Concepto: Política

- **La política:** mapeo entre los estados del entorno percibidos por el agente y las acciones que el agente realizará cuando alcance cada uno de esos estados.
- Se suele representar con la letra griega π y habrá tantas como combinaciones de acciones y estados haya
- El aprendizaje consiste en encontrar las **política optima** π^* de entre todas las posibles
- **Value Iteration:** Permite encontrar la política óptima si conocemos la distribución subyacente del entorno.
- Al finalizar el Value Iteration sabemos cuales son los **valores óptimos** V^* de cada estado:

$$\pi^* =_{a \in A} \sum_{s' \in S} T(s, a, s')(R(s, a, s')) + \gamma V^*(s')$$

Formalización del estado: Asumiendo Markov

Un estado contiene información útil sobre la historia:

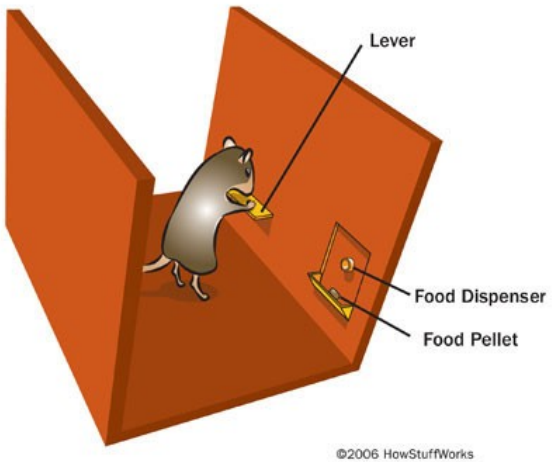
Definición

Un estado S_t es un estado Markoviano si y solo si

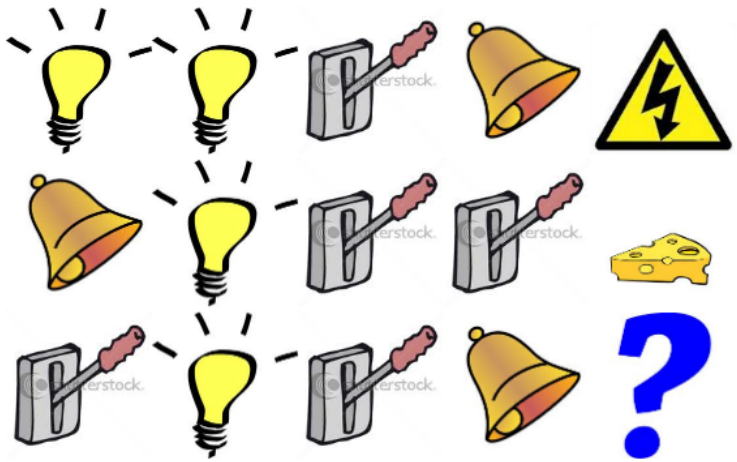
$$P[S_{t+1}|S_t] = P[S_{t+1}|S_1, S_2, ...S_t]$$

El futuro es independiente del pasado dado el presente (el estado actual)

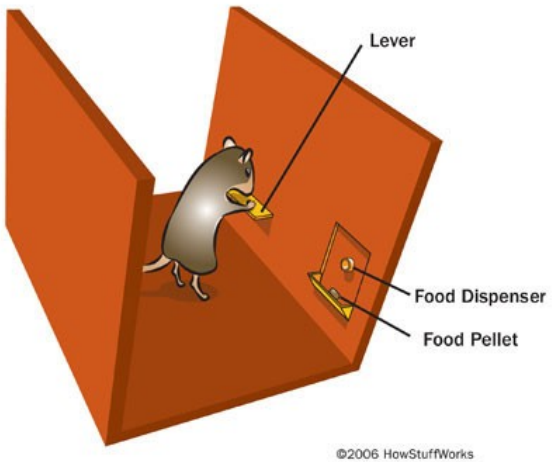
El Estado



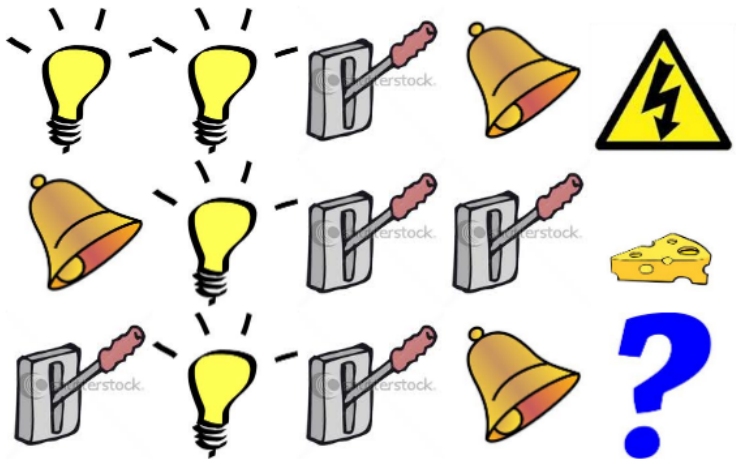
Skinner Box



El Estado

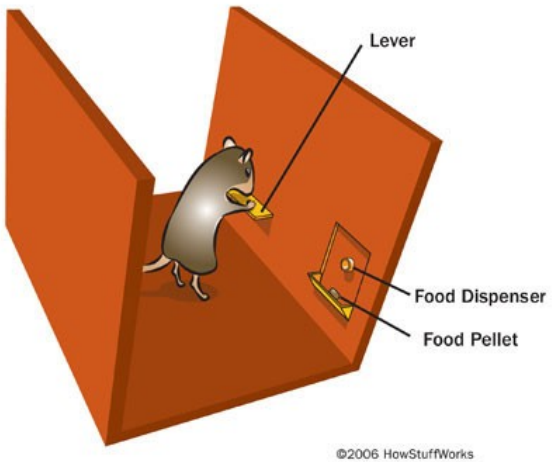


Skinner Box

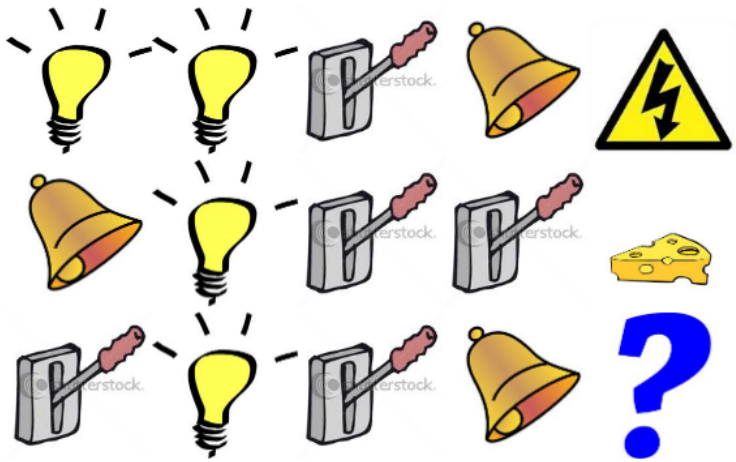


- ¿Si el estado = los 3 últimos elementos?

El Estado

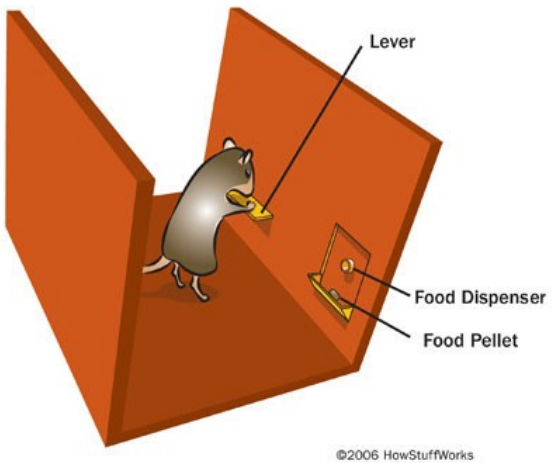


Skinner Box

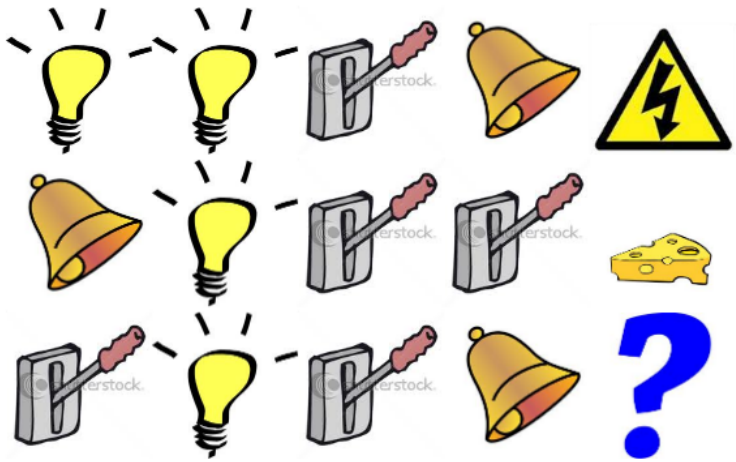


- ¿Si el estado = los 3 últimos elementos?
- ¿Si el estado = contadores de luces, campanas y palancas?

El Estado



Skinner Box



- ¿Si el estado = los 3 últimos elementos?
- ¿Si el estado = contadores de luces, campanas y palancas?
- ¿Si el estado = la secuencia completa?

Problemas con los MDPs

No se suele disponer de las **funciones de transición**.
Solo disponemos de los estados y de las acciones y el premio que está asociado a transicionar de un determinado estado a otro.
Dos estrategias posibles:

- **Model based:** Consiste en aprender las funciones de transición y del premio y luego aplico value iteration. No se suele emplear porque es muy costoso.
- **Model free:** Consiste en aprender el valor de cada acción a través de muestras o episodios. **Aprendizaje temporal por episodios.**

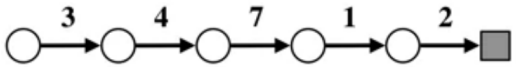
Aprendizaje temporal por episodios

Un **episodio** consiste en exponer al agente a **un ciclo completo** donde hay un estado inicial y un estado final. El episodio está formado por:

- Una lista de estados
- Acciones (posibles acciones a partir de un estado)
- Premios
- Nuevos estados (posibles estados a partir de un estado)

Aprendizaje por episodios: Métodos

- **Monte Carlo:** El premio se contabiliza al final del episodio. Los estados se recorren en orden inverso y así los premios se van acumulando en orden inverso.



$\gamma = 0.5$

$$G_0 = R_1 + \gamma\ R_2 + \gamma^2 R_3 + \gamma^3 R_4 + \gamma^4 R_5 = 7$$
$$G_1 = R_2 + \gamma\ R_3 + \gamma^2 R_4 + \gamma^3 R_5 = 8$$
$$G_2 = R_3 + \gamma\ R_4 + \gamma^2 R_5 = 8$$
$$G_3 = R_4 + \gamma\ R_5 = 2$$
$$G_4 = R_5 = 2$$
$$G_5 = 0$$

- **Aprendizaje Temporal:** No se espera hasta el final. En cada paso se va actualizando el valor del estado haciendo una media ponderada entre el valor actual y lo que le propone el "futuro".

$$V(S_{t+1}) = (1-\alpha)V(S_t) + \alpha[R_{t+1} + \gamma V(S'_t))]$$
$$V(S_{t+1}) = V(S_t) + \alpha[R_{t+1} + \gamma V(S'_t) - V(S_t)]$$

Aprendizaje temporal por episodios

- SARSA: Como Q-learning salvo en vez de seleccionar el valor Q de la mejor acción en s' se selecciona el valor Q de una acción seleccionada según la política (epsilon greedy,..)
- Q-Learning[2]

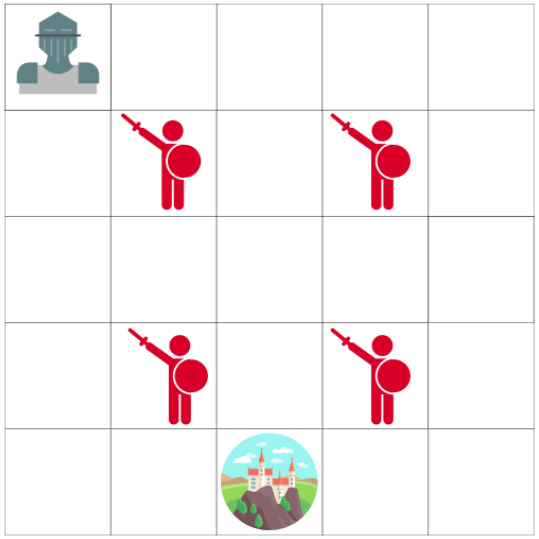
Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$

Loop for each episode:
  Initialize S
  Loop for each step of episode:
    Choose A from S using policy derived from Q (e.g., ε-greedy)
    Take action A, observe R, S'
    $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma \max_a Q(S', a) - Q(S, A)\big]$
    $S \leftarrow S'$
  until S is terminal

[2] ¡¡IMPORTANTE!! También lo vereis escrito así
$Q(S, A) \leftarrow (1 - \alpha)Q(S, A) + \alpha[R + \gamma max_a Q(S', a)]$

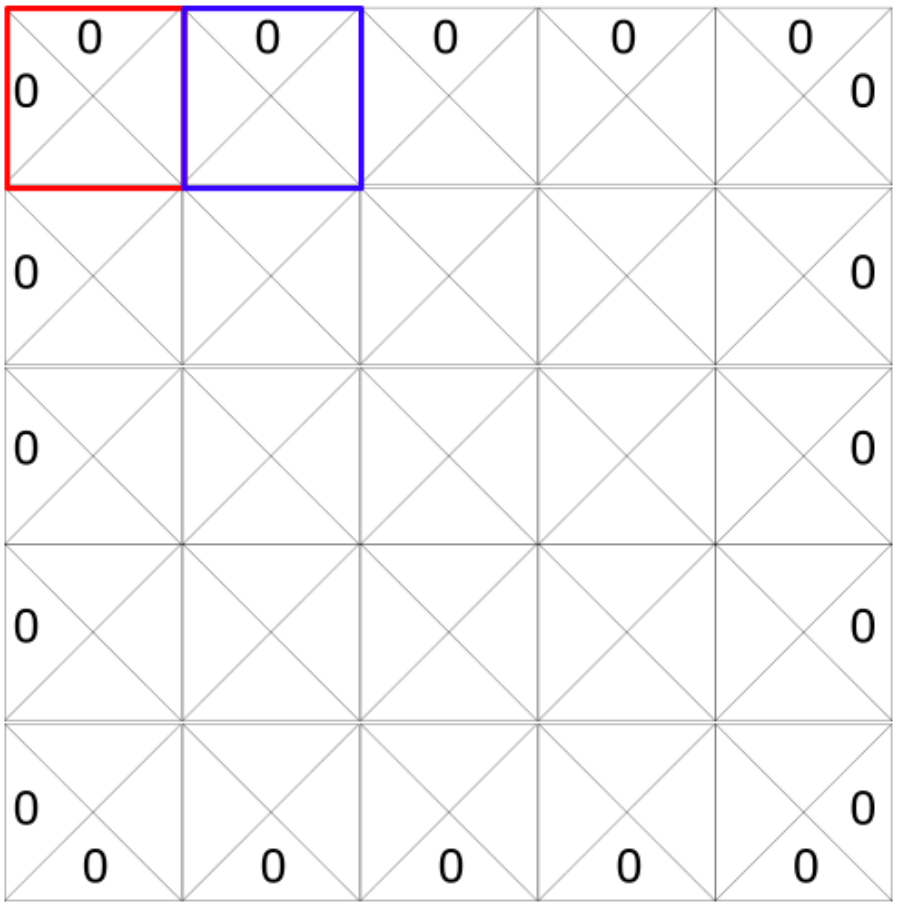
Aprendizaje temporal por episodios Q-learning (Q-table). Ejemplo [3]



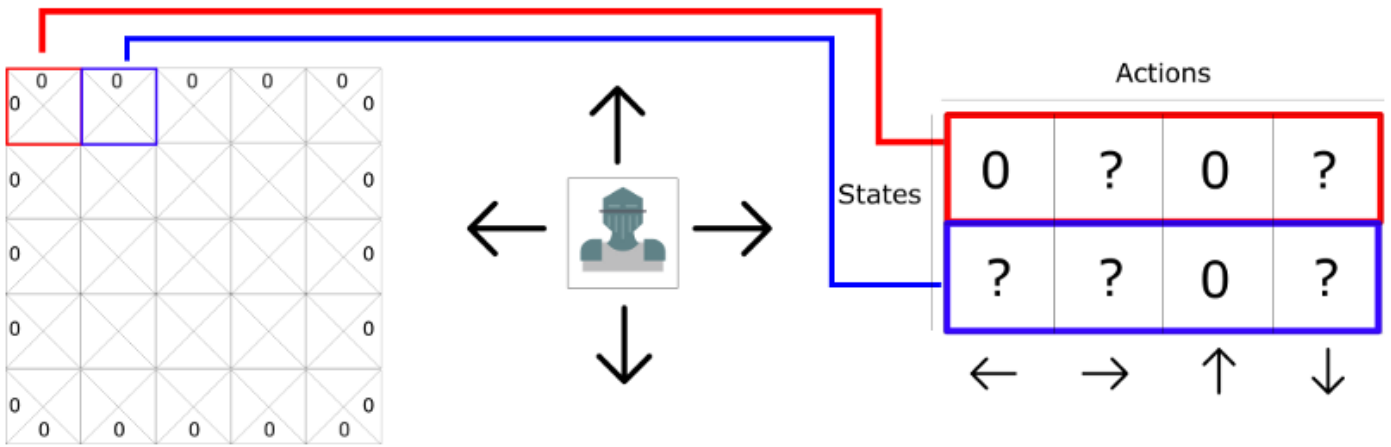
- Cada paso es un premio de -1 (para indicar que el camino más largo es peor).
- Si tocas a un enemigo el premio es -100 y el episodio finaliza.
- Si estás en el castillo el premio es +100.

[3] Fuente: https://www.freecodecamp.org/news/diving-deeper-into-reinforcement-learning-with-q-learning-c18d0db58efe/

Aprendizaje temporal por episodios Q-learning (Q-table). Ejemplo de un gridworld

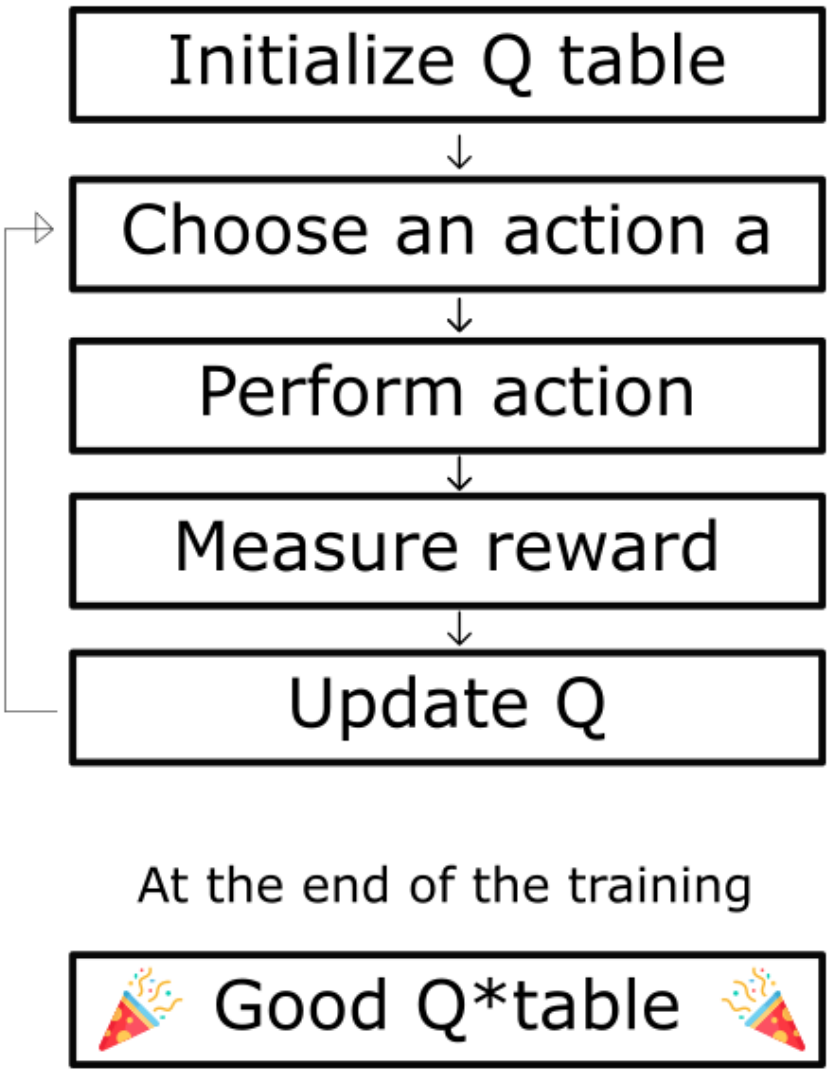


Aprendizaje temporal por episodios Q-learning (Q-table). Ejemplo

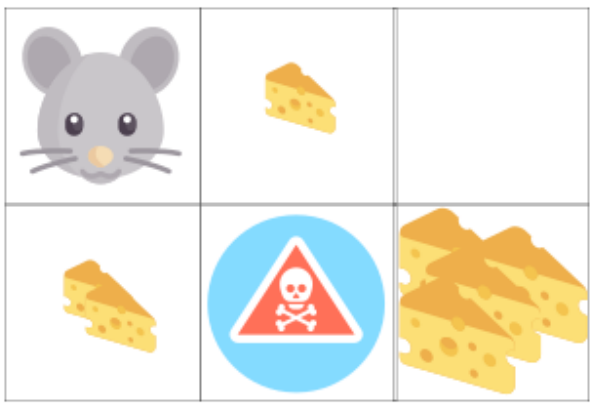


$$Q^{\pi}(s_t, a_t) = \underline{E}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + ...|s_t, a_t]$$

Aprendizaje temporal por episodios Q-learning (Q-table). Ejemplo



Ejercicio de ejemplo



	←	→	↑	↓
Start	0	0	0	0
Small cheese	0	0	0	0
Nothing	0	0	0	0
2 small cheese	0	0	0	0
Death	0	0	0	0
Big cheese	0	0	0	0

Aprendizaje temporal por episodios

Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$

Loop for each episode:
  Initialize S
  Loop for each step of episode:
    Choose A from S using policy derived from Q (e.g., ε-greedy)
    Take action A, observe R, S'
    $Q(S, A) \leftarrow Q(S, A) + \alpha\left[R + \gamma \max_a Q(S', a) - Q(S, A)\right]$
    $S \leftarrow S'$
  until S is terminal

¿Cómo equilibramos la exploración versus la explotación?

- Ir decrementando el ϵ según pasan los episodios
- Añadir una función de exploración que modifica las actualización ligeramente añadiendo un "bias" sobre las acciones aun no experimentadas
  $Q(S', a) \rightarrow Q(S', a) + k/(n+1)$ donde $n+1$ es el número de veces que se ha ejecutado esa acción.

Limitaciones del Q-Learning

Con configuraciones y un número de estados pequeño como los de los ejemplos anteriores, el algoritmo funcionará. Pero,

- ¿qué sucede con casos como el del Pacman en el que el número de estados es enorme?¿es realista pensar que los vamos a poder explorar todos?
- solución: quizás pueda jugar con la representación de los estados

Q-Learning Aproximado

Queremos encontrar una representación que generalize y permita agrupar casos similares[4]

[4] Ejemplo de Dan Klein

Q-Learning Aproximado

Solución: Representación de estados como vector de rasgos

- Distancia al fantasma más cercano
- Distancia al punto más próximo
- Número de fantasmas
- $\frac{1}{(dist.AlPunto)^2}$
- ...

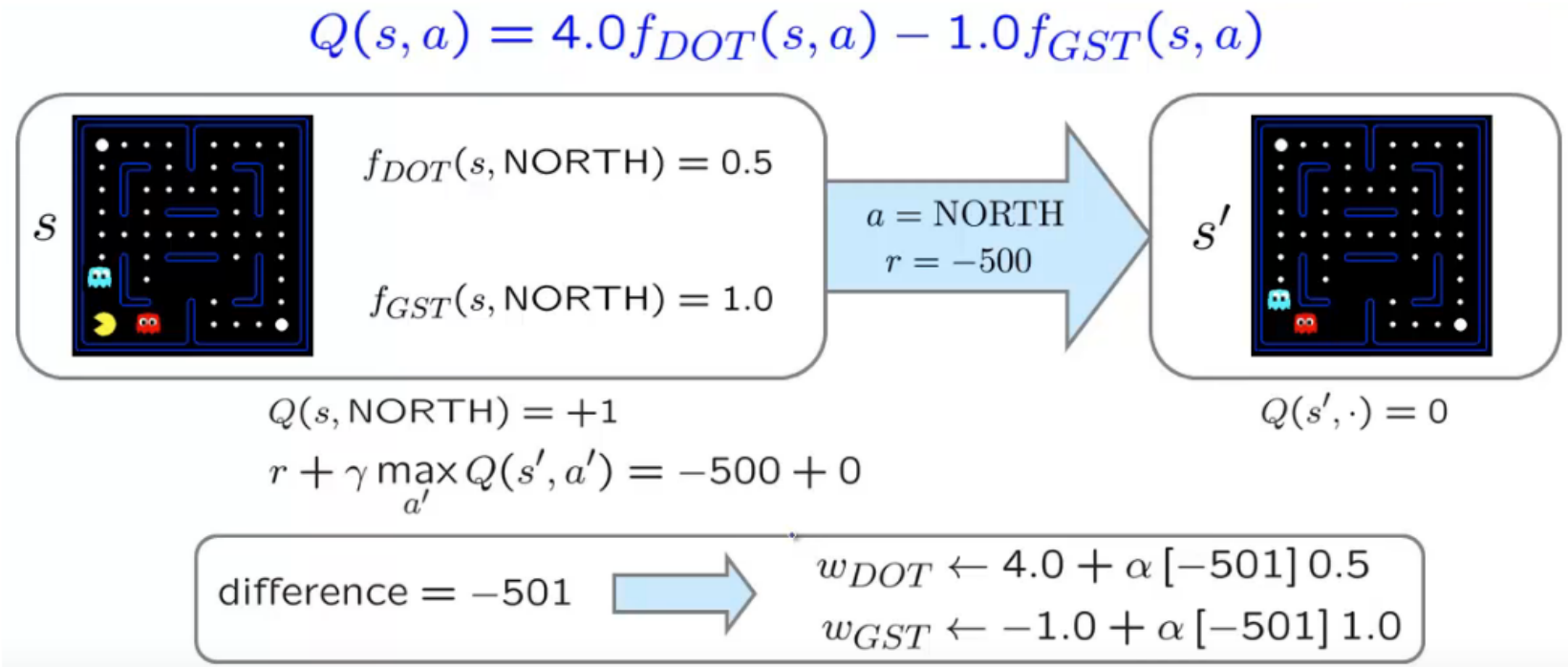
Q-Learning Aproximado

$$Q(S,A) = Q(S,A) + \alpha[(R + \gamma max'_a Q(S',a')) - Q(S,A)]$$

- $Q(S,A) = w_1 f_1(S,A) + w_1 f_1(S,A) + ... + w_n f_n(S,A)$
- transición (S,A,S',R)
  - diferencia entre:
    - $R + \gamma max'_a Q(S',a')$: El futuro si se realiza la acción
    - Q(S,A): mi estado actual

$$Q(S,A) = Q(S,A) + \alpha[diferencia]$$
$$w_1 = w_1 + \alpha[diferencia] f_1(S,A)$$

Q-Learning Aproximado[5]



[5] Ejemplo de Dan Klein

Bibliografía

- Reinforcement Learning, An Introduction (Second Edition). By Richard S. Sutton and Andrew G. Barto `https://web.stanford.edu/class/psych209/Readings/SuttonBartoIPRLBook2ndEd.pdf`
- Berkeley curso de Inteligencia Artificial (Dan Klein)
- DLRL2019 (Adam White): `https://www.youtube.com/watch?v=RancMV1wECg`