Aprendizaje Automático
Segundo Cuatrimestre de 2016

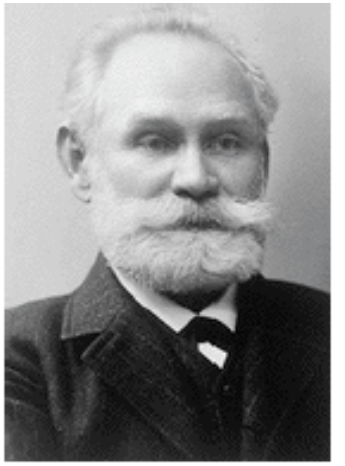
Aprendizaje por Refuerzos

Gracias a Carlos “Greg” Diuk por los materiales para las transparencias.

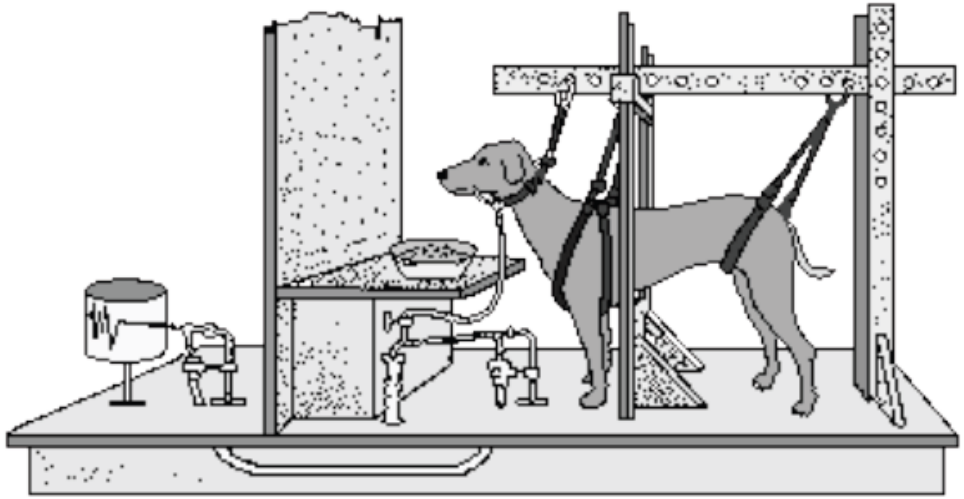
DEPARTAMENTO
DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

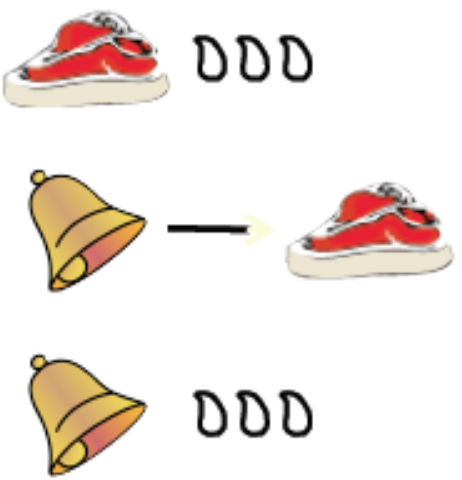
Aprendizaje por Condicionamiento

- Primeras teorías:
  - Condicionamiento clásico o Pavloviano.



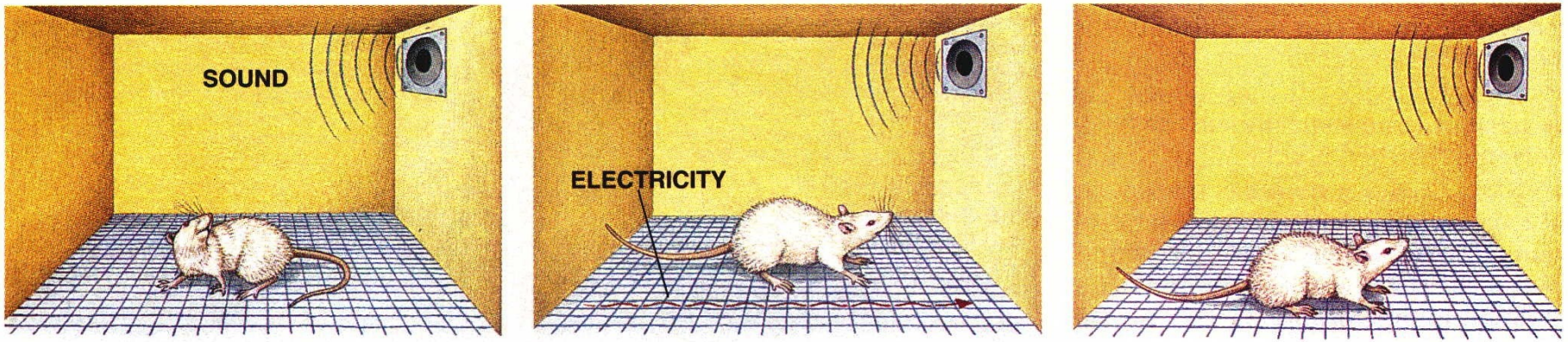
Ivan Pavlov
(1849-1936)

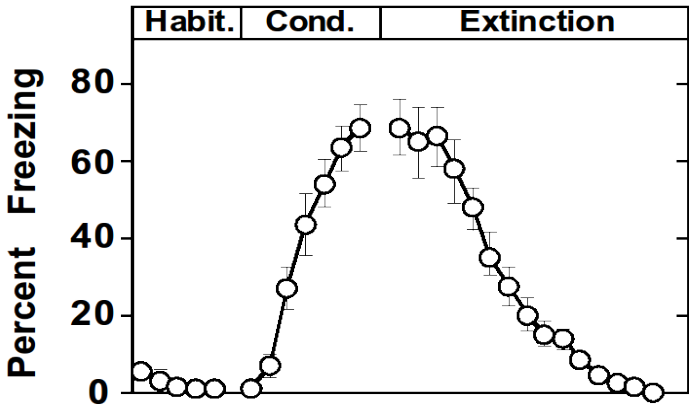




Aprendizaje por Condicionamiento

Habituación (tono) | Condicionamiento (tono+shock) | Extinción (tono)





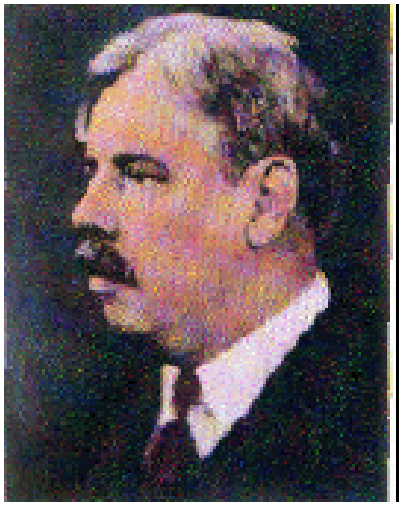
Crédito: Quirk lab, U de Puerto Rico

- Visión ultra-limitada del aprendizaje, pero funciona! Demuestra que los animales pueden aprender relaciones arbitrarias entre estímulo → respuesta.

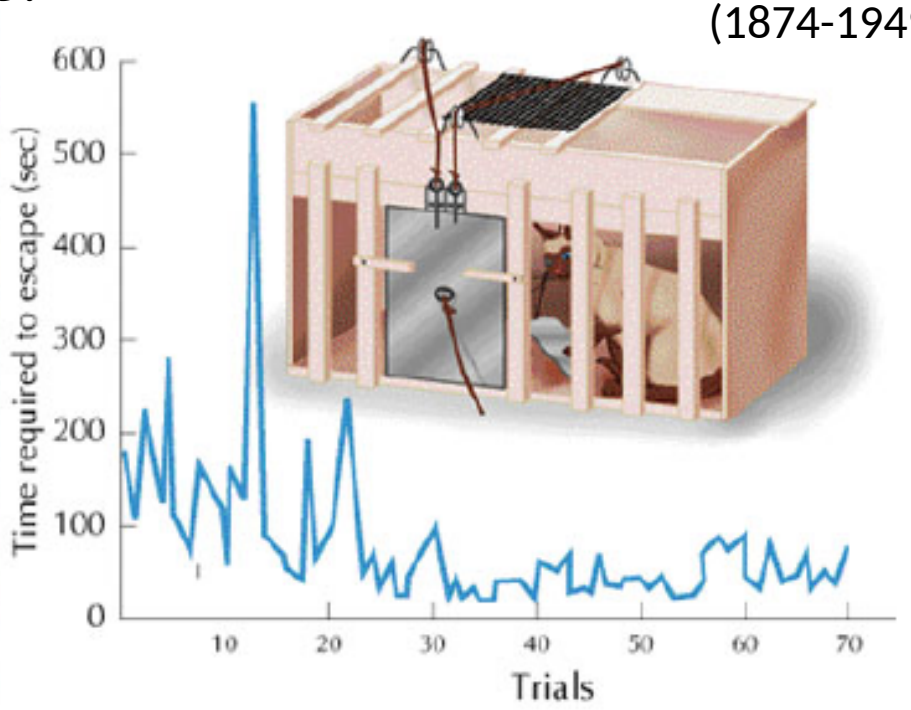
- Pero el perro de Pavlov no tomó decisiones.
- Hasta ahora no tuvo que hacer nada.
  - Sólo salivó porque la campana le recordó el churrasco.
- Agreguemos "control": no solamente estímulo-respuesta.
- Las acciones que tomamos tienen consecuencias.

Condicionamiento Instrumental/Operacional

- Thorndike experimentó con gatos hambrientos tratando de escapar de una jaula.
- Midió el "tiempo requerido para escapar" como métrica de aprendizaje.
- "Curva de aprendizaje"



Edward Thorndike (1874-1949)



Crédito: Yael Niv

Condicionamiento Instrumental/Operacional

- La lección importante a extraer:

Los animales no sólo pueden aprender relaciones estímulo-respuesta arbitrarias, sino también comportamientos arbitrarios en base a dichos estímulos.

Crédito: Björn Brembs, FU Berlin

Rescorla & Wagner (1972)

- Aprendizaje guiado por errores: el cambio en el *valor* de una asociación es proporcional a la diferencia entre nuestra predicción y lo observado:

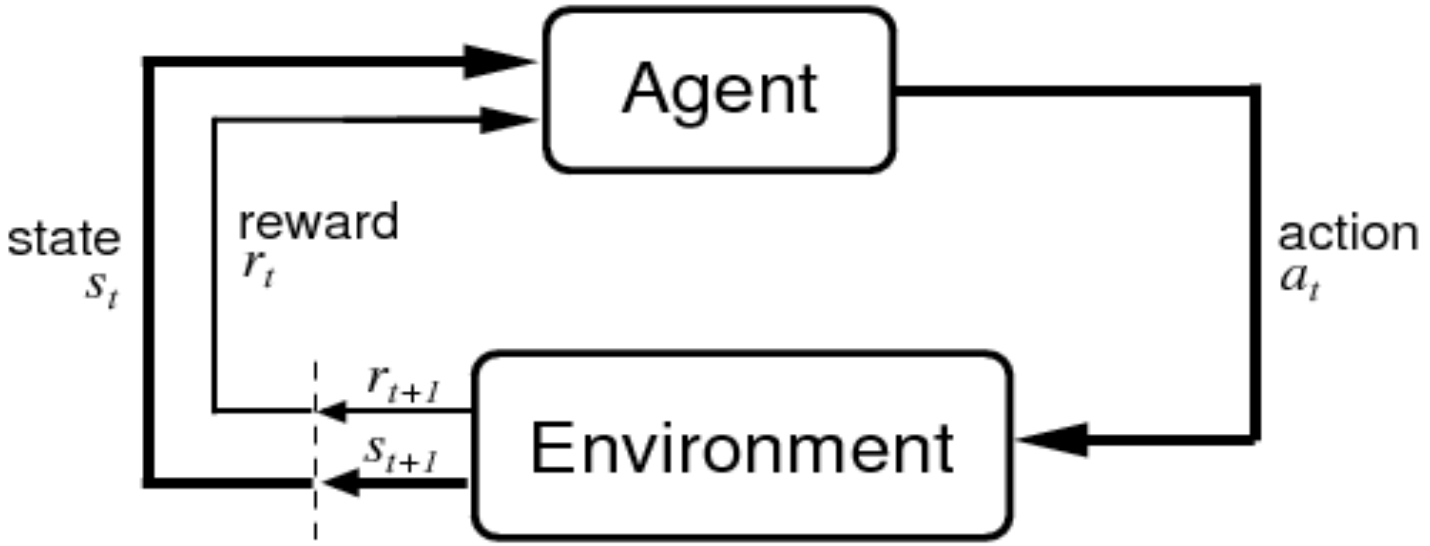
$$V_{\text{nuevo}} = V_{\text{viejo}} + \alpha\,(R - V_{\text{viejo}})$$

tasa de aprendizaje

error de predicción

Aprendizaje por Refuerzos

- Agente (ej, un robot):
  - Tiene sensores para observar el estado de su entorno.
  - Puede realizar acciones para alterar ese estado.
  - Cada acción conlleva una recompensa numérica inmediata.



Crédito: Sutton y Barto

Aprendizaje por Refuerzos

- Tarea del agente: aprender una estrategia o política de control para elegir las acciones que maximicen las recompensas.

$$\pi : S \rightarrow A$$

- Diferencias con aprendizaje supervisado:
  - Las recompensas vienen con demora (ej: ganar un juego te premia al final). Un agente debe aprender a discernir cuáles acciones de una secuencia fueron las meritorias.
  - Exploración: la distribución de los datos de entrenamiento está determinada por las acciones que el agente elige.

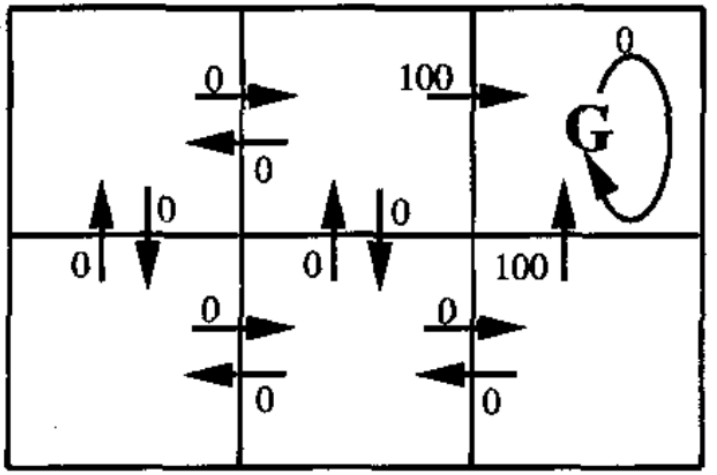
Formalismo estándar

Proceso de Decisión de Markov (MDP)

- Conjunto de estados S
- Conjunto de acciones A
- Función de transición $T : S \times A \rightarrow S$
- Función de recompensa $R : S \times A \rightarrow \mathbb{R}$

- Ejemplo: *Grid World*

  6 estados, 5 acciones ($\leftarrow \uparrow \rightarrow \downarrow \circlearrowright$)



Tom Mitchell: *Machine Learning*, p372

Proceso de Decisión de Markov (MDP): $\langle S, A, T(s,a), R(s,a) \rangle$

Definimos:

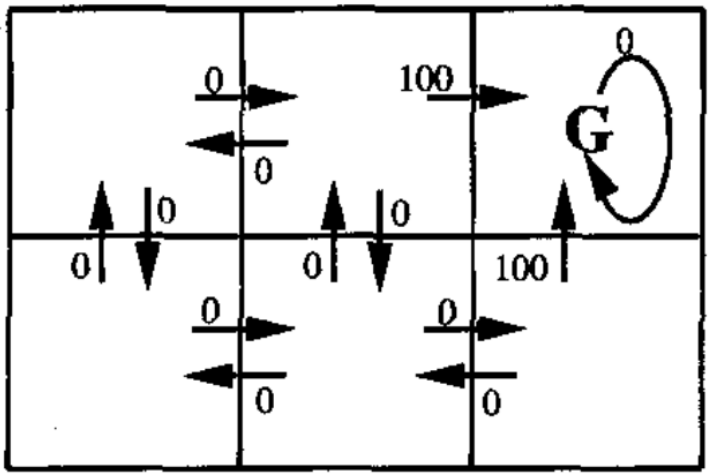
- Política $\pi : S \rightarrow A$

- Función de valor:

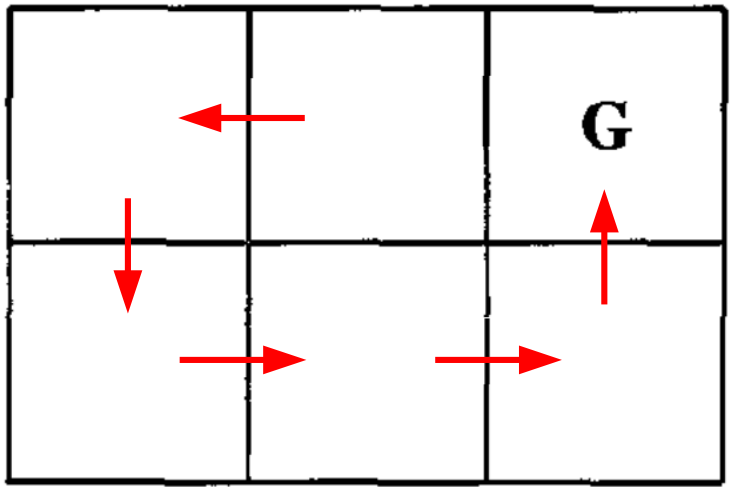
$$V^{\pi}(s_t) = r_{t+1} + \gamma\, r_{t+2} + \gamma^2 r_{t+3} + \gamma^3 r_{t+4} + \ldots = \sum_{i=1}^{\infty} \gamma^{i-1} r_{t+i}$$

- $V^{\pi}(s_t)$ es el valor acumulado que se consigue al seguir la política π a partir del estado s_t.
- γ se conoce como factor de descuento ($0 \leq \gamma < 1$).

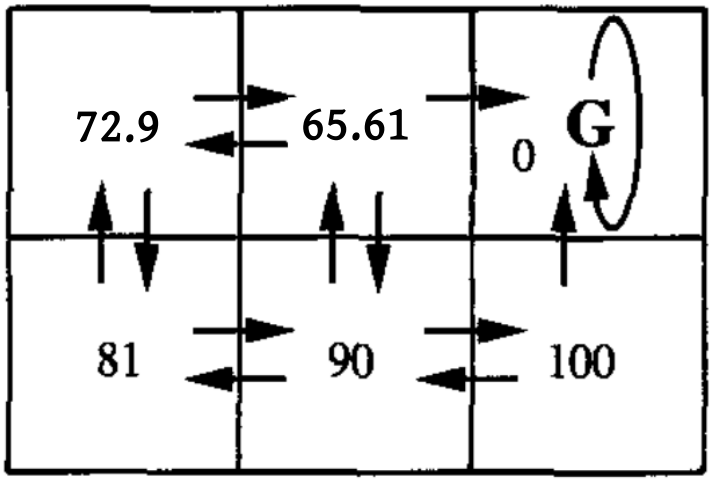
Función de recompensa *R*:



Política π:



Función de valor V^{π}:



Tom Mitchell: *Machine Learning*, p372

- Función de valor:

$$V^{\pi}(s) = r_0 + \gamma\, r_1 + \gamma^2 r_2 + \ldots = \sum_{t=0}^{\infty} \gamma^t r_t$$

- Definición recursiva (ecuación de Bellman):

$$V^{\pi}(s) = \underbrace{R(s, \pi(s))}_{\text{recompensa inmediata}} + \gamma \underbrace{V^{\pi}(T(s, \pi(s)))}_{\text{valor del siguiente estado}}$$

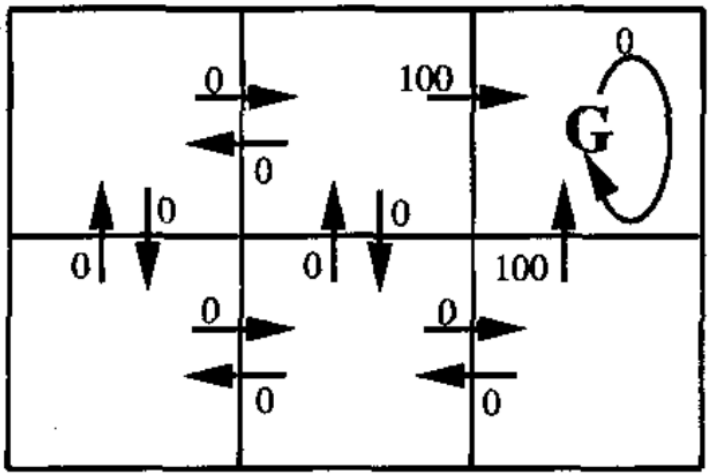
- Objetivo: aprender la función de valor óptima V*, fruto de ejecutar la política óptima π^*:

$$\pi^*(s) = \underset{a}{\operatorname{argmax}} \left[R(s,a) + \gamma\, V^*(T(s,a)) \right]$$

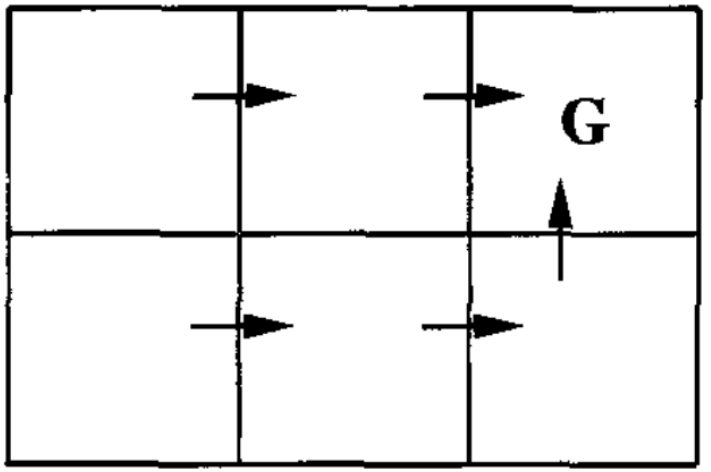
$$V^*(s) = \max_{a} \left[R(s,a) + \gamma\, V^*(T(s,a)) \right]$$

Idea para un algoritmo iterativo de aprendizaje

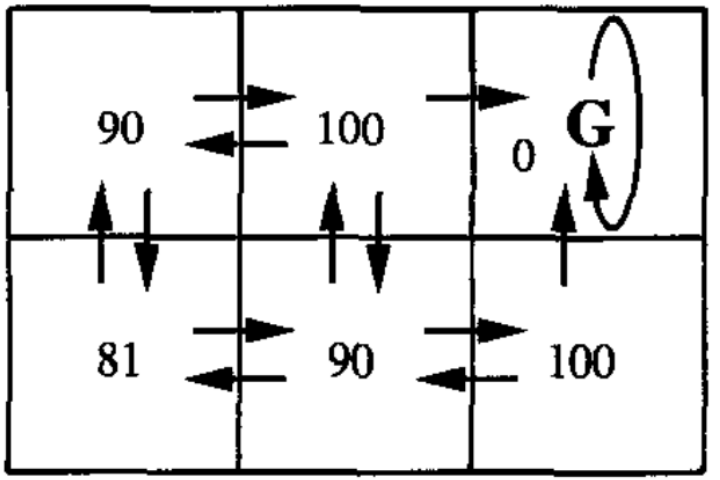
Función de recompensa *R*:



Política óptima π*:



Función de valor *V**:



Tom Mitchell: *Machine Learning*, p372

Algoritmo 1: *Value Iteration*

$V(s) \leftarrow 0 \;\; \forall s \in S$
$\Delta \leftarrow 0$
Repetir
    Para cada $s \in S$**:**
        $V(s) \leftarrow \max_a \left[R(s,a) + \gamma\, V(T(s,a))\right]$
    Fin para
Hasta que el modelo sea "suficientemente bueno"

Definimos $\pi(s) = \underset{a}{\operatorname{argmax}} \left[R(s,a) + \gamma\, V(T(s,a))\right]$

- VI en acción: http://www.cs.ubc.ca/~poole/demos/mdp/vi.html
- VI requiere conocer el modelo: *S*, *A*, *R* y *T* (es "*model-based*").

- Objetivo: aprender la función de valor óptima V*, fruto de ejecutar la política óptima π^*:

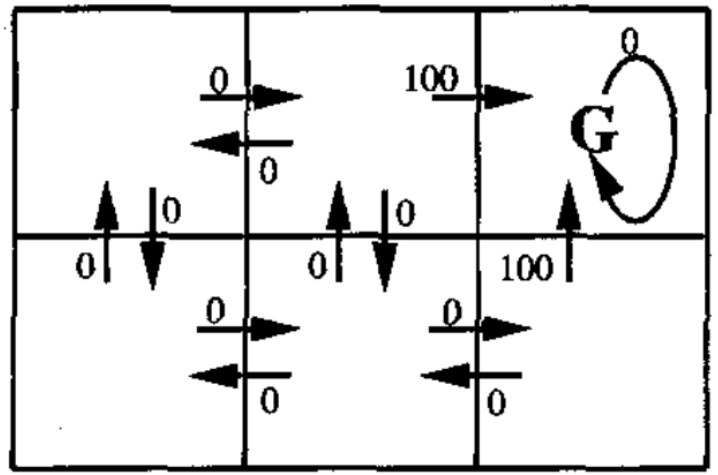
$$\pi^*(s) = \underset{a}{\operatorname{argmax}} \left[R(s,a) + \gamma \, V^*(T(s,a)) \right]$$

$$V^*(s) = \max_a \underbrace{\left[R(s,a) + \gamma \, V^*(T(s,a)) \right]}_{\text{Def: } Q(s,a)}$$

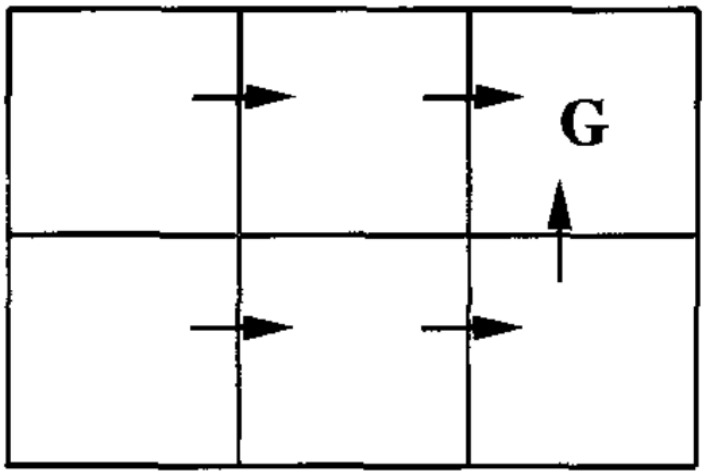
- *Q(s, a)* es la máxima ganancia esperada a partir de *s*, ejecutando la acción *a*.
- Si expandimos el término *V*(T(s,a))* en la definición de *V*(s)*, llegamos a esta definición recursiva de *Q(s, a)*:

$$Q(s,a) = R(s,a) + \gamma \max_{a'} Q(T(s,a), a')$$

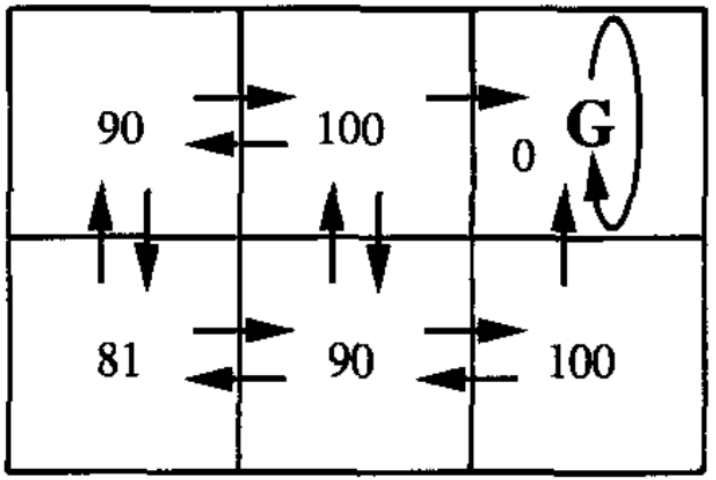
Función de recompensa *R*:



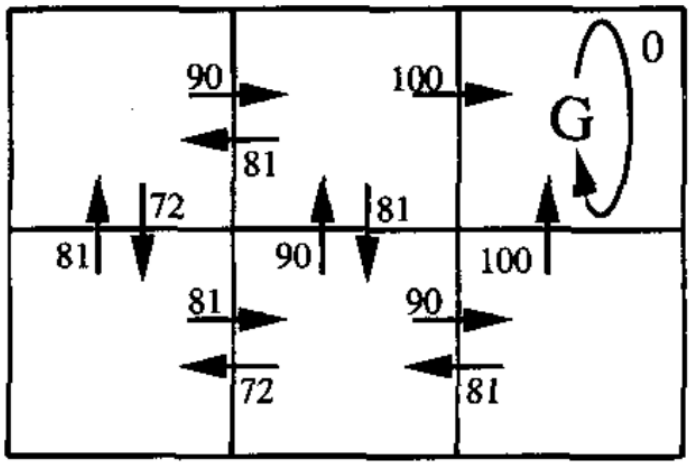
Política óptima π^*:



Función de valor V^*:



$Q(s, a)$:



Tom Mitchell: *Machine Learning*, p372

$$Q(s,a) = R(s,a) + \gamma \max_{a'} Q(T(s,a), a')$$

Idea para otro algoritmo iterativo de aprendizaje:

$$Q(s,a) \leftarrow R(s,a) + \gamma \max_{a'} Q(T(s,a), a')$$

Si incorporamos una tasa de aprendizaje α $(0 < \alpha \leq 1)$:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma \max_{a'} Q(s', a') - Q(s,a) \right]$$

Algoritmo 2: *Q-Learning*

Inicializar $Q(s, a)$ al azar
Repetir (para cada episodio):
  Inicializar s
  Repetir (para cada paso del episodio):
    Elegir a en s según una política basada en Q
    Ejecutar la acción a, observar r, s'
    $Q(s, a) \leftarrow Q(s, a) + \alpha \left[r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right]$
    $s \leftarrow s'$
  Hasta que s sea terminal

Definimos $\pi(s) = \underset{a}{\operatorname{argmax}} \left[Q(s, a) \right]$

- QL en acción: http://www.cs.ubc.ca/~poole/demos/rl/q.html
- QL no requiere conocer el modelo (es "*model-free*").
- Falta decidir cómo elegir la acción *a* en el estado *s*.

K-armed bandits

Cada máquina tiene su propia probabilidad de pagar.
¿Cómo hacemos para maximizar nuestra ganancia?

Dilema exploración-explotación: cuándo explorar mejores opciones, cuándo explotar lo que ya sabemos.

Bandidos: Eligiendo un brazo

- Estrategia *ε-greedy*:
  - Con prob. ε: elegir al azar (distribución uniforme).
  - Con prob. (1–ε): elegir el mejor brazo conocido.
- Estrategia *ε-first* (primero exploración, después explotación).
  - Primeros (1–ε)% trials: elegimos al azar (distrib. uniforme).
  - Restantes ε% trials: elegimos el mejor brazo conocido.
- Estrategia *softmax*:
  - La probabilidad de elegir *a* es proporcional al valor *Q*(*s*,*a*):

$$\Pr(a) = \frac{e^{Q(s,a)/\tau}}{\sum_a e^{Q(s,a)/\tau}}$$

$\tau > 0$ es la temperatura.
τ alta: elección al azar (distr. uniforme)
τ se reduce gradualmente.

Resumen

- Proceso de Decisión de Markov (MDP):

$$\langle S, A, T(s,a), R(s,a) \rangle$$

- Política π, función de valor V, función Q.
- Algoritmos *Value Iteration* y *Q-Learning*.
- *K-armed bandits*. Estrategias *ε-greedy*, *ε-first*, *softmax*.