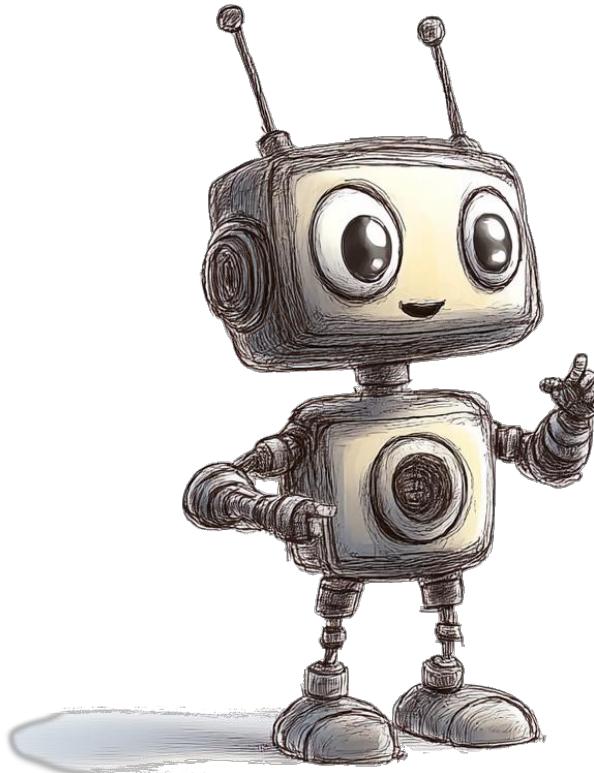


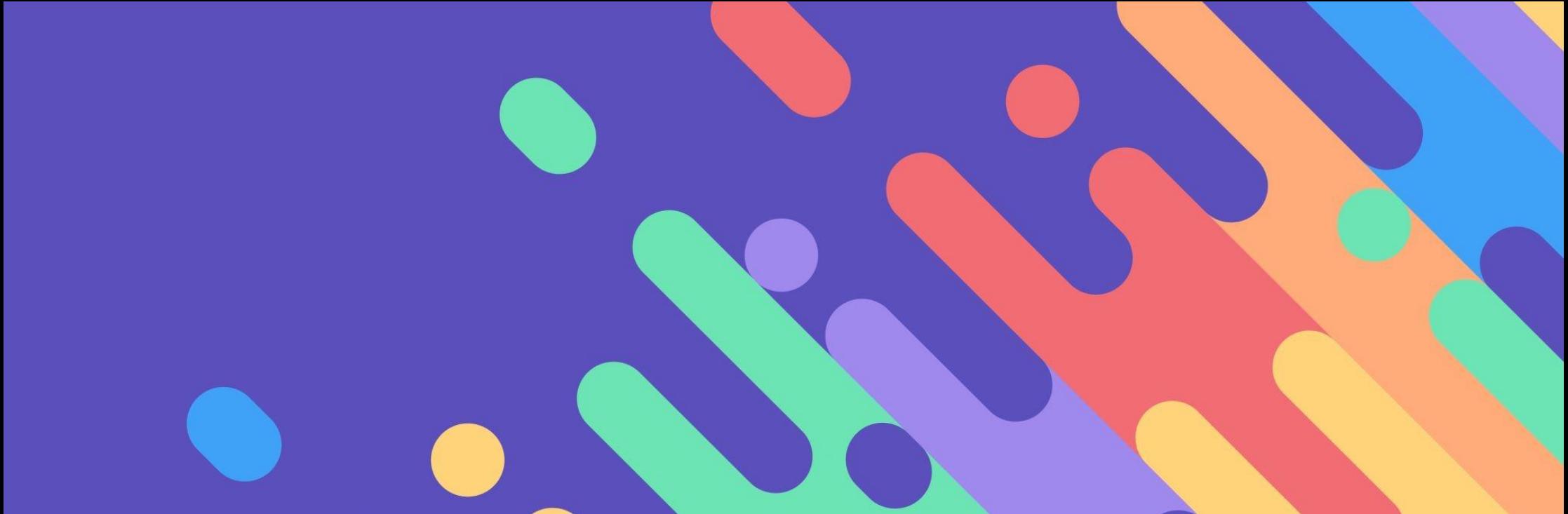
---

# APRENDIZAJE POR REFUERZO



Los temas que veremos en este video son:

- Definición de aprendizaje por refuerzo
- Proceso de decisión de Márkov
- Ecuación de Bellman
- Estrategias de aprendizaje
- Q-Learning



---

# APRENDIZAJE POR REFUERZO

# APRENDIZAJE POR REFUERZO

Definamos el **Aprendizaje por refuerzo**...

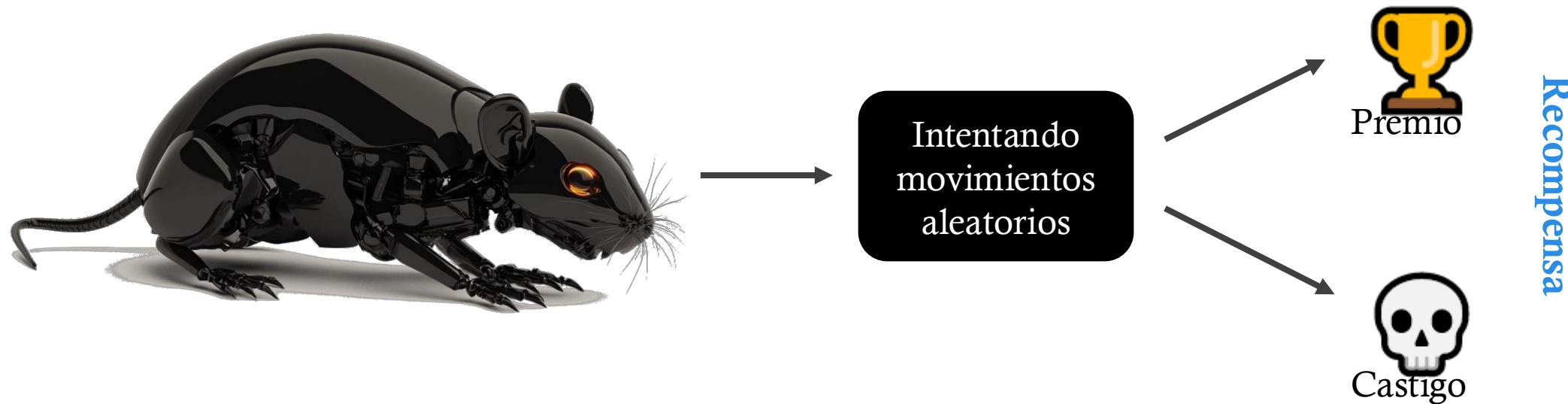
*El aprendizaje por refuerzo es una rama del aprendizaje automático en la que un agente aprende a tomar decisiones óptimas al interactuar con un entorno, con el objetivo de maximizar una señal de recompensa acumulada a lo largo del tiempo.*



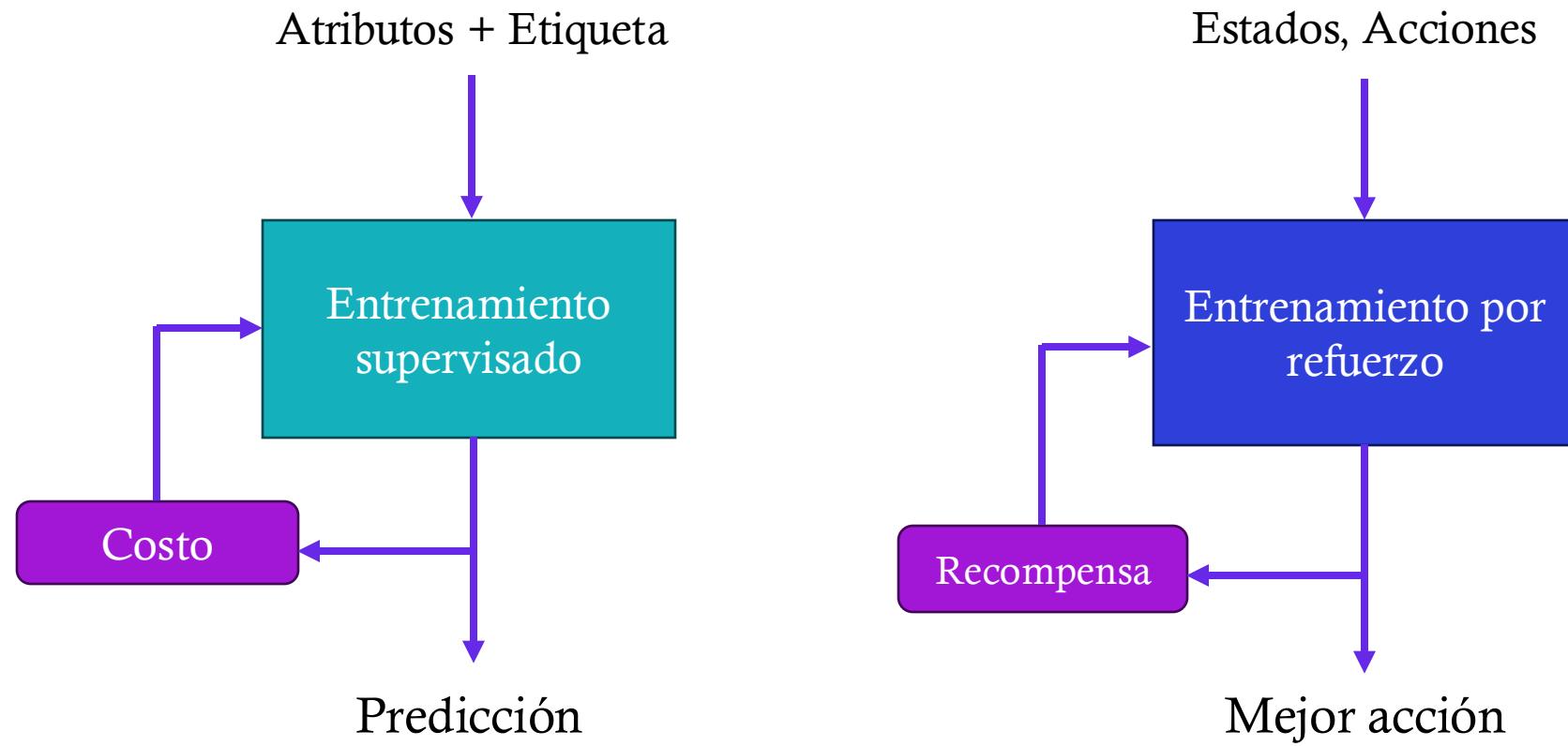
# APRENDIZAJE POR REFUERZO

Un agente puede aprender a jugar al ajedrez mediante aprendizaje supervisado, recibiendo ejemplos de situaciones de juego junto con los mejores movimientos para cada una.

Pero, ¿qué pasa si no hay un profesor que le indique qué hacer?

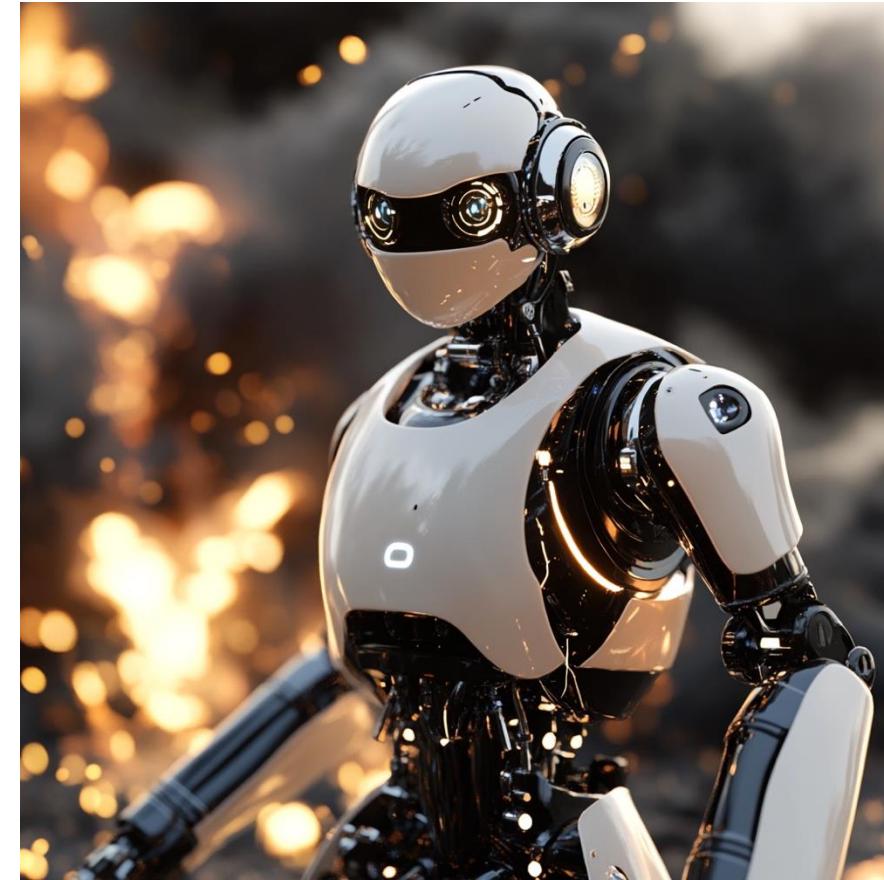


# APRENDIZAJE POR REFUERZO



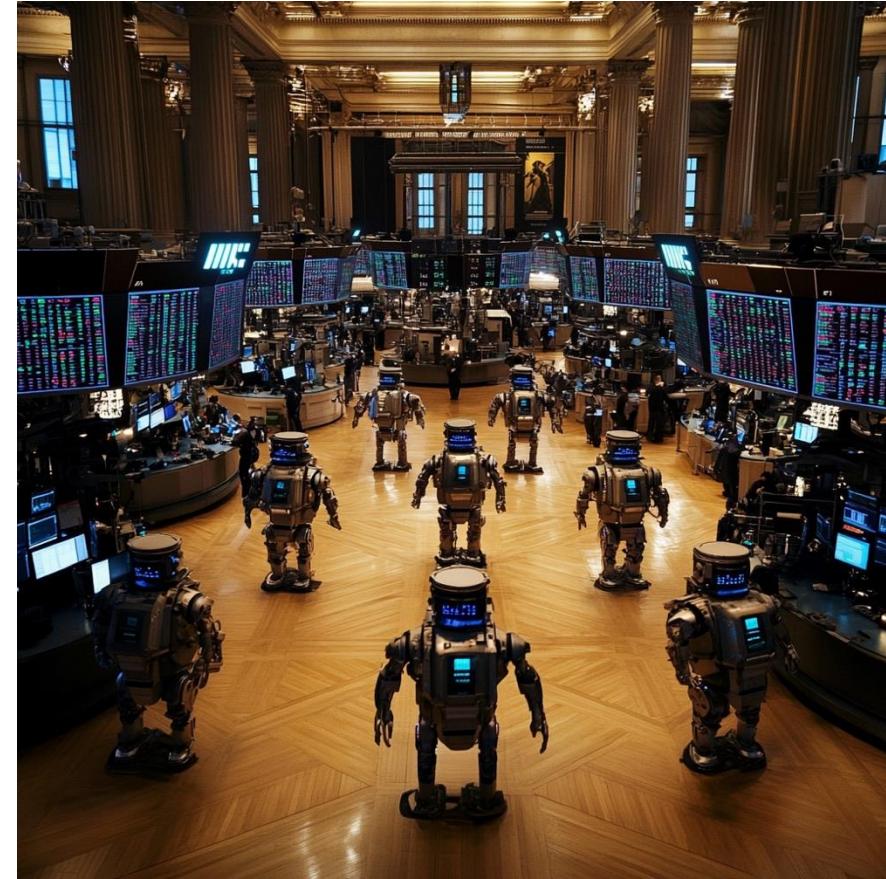
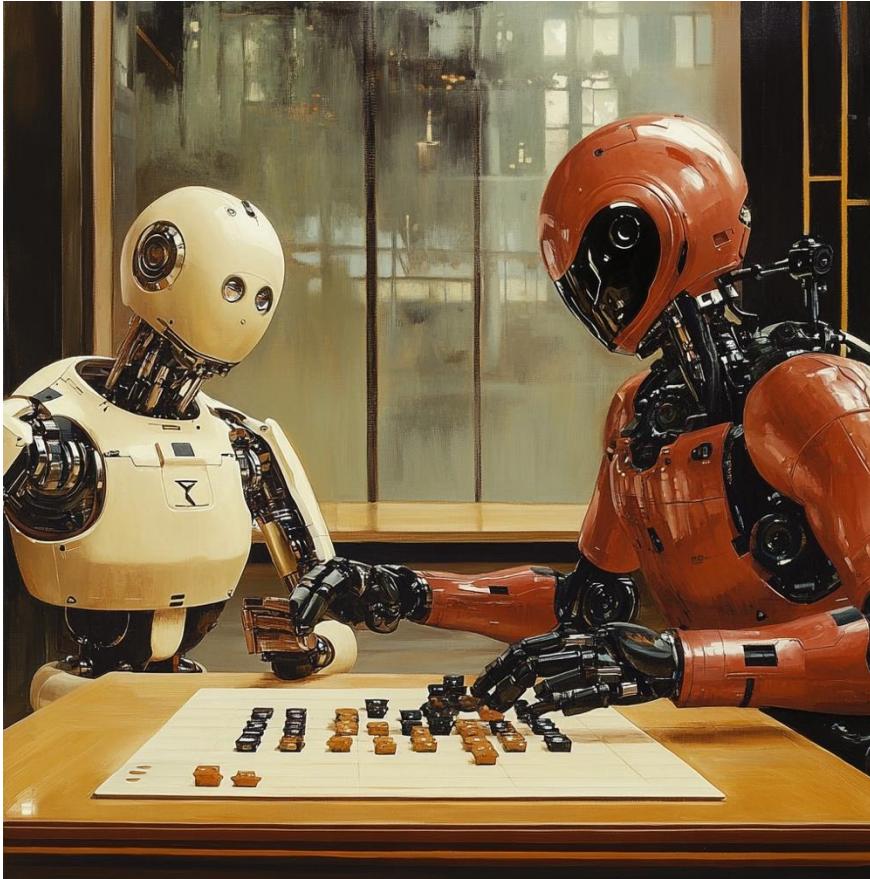
---

# APRENDIZAJE POR REFUERZO



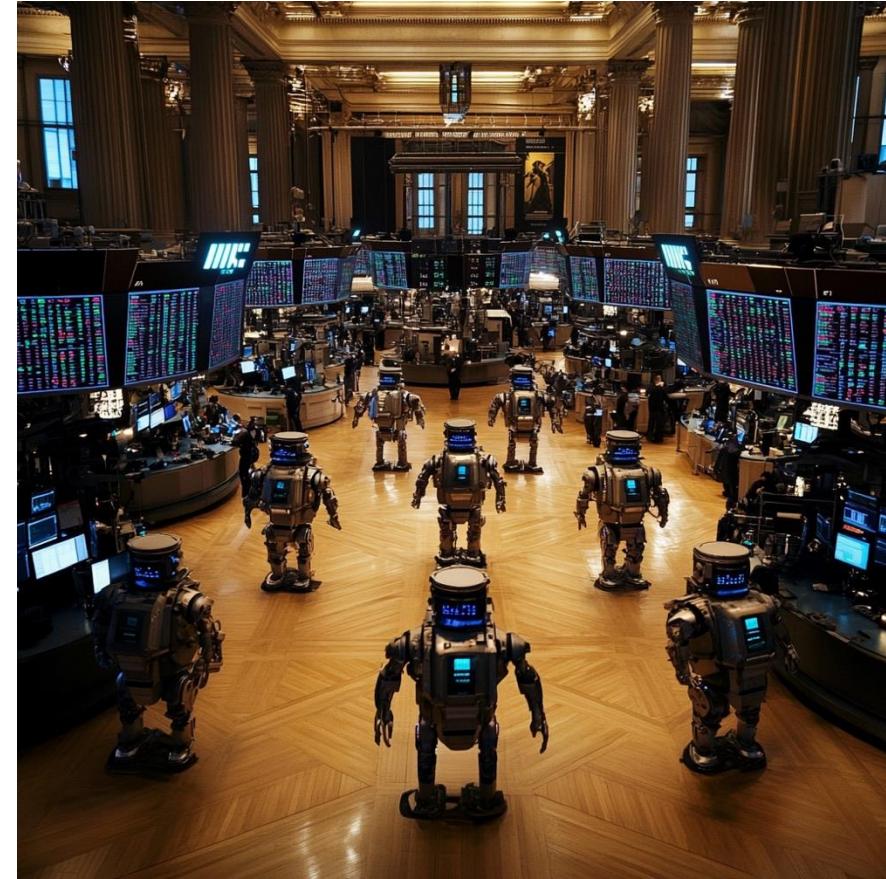
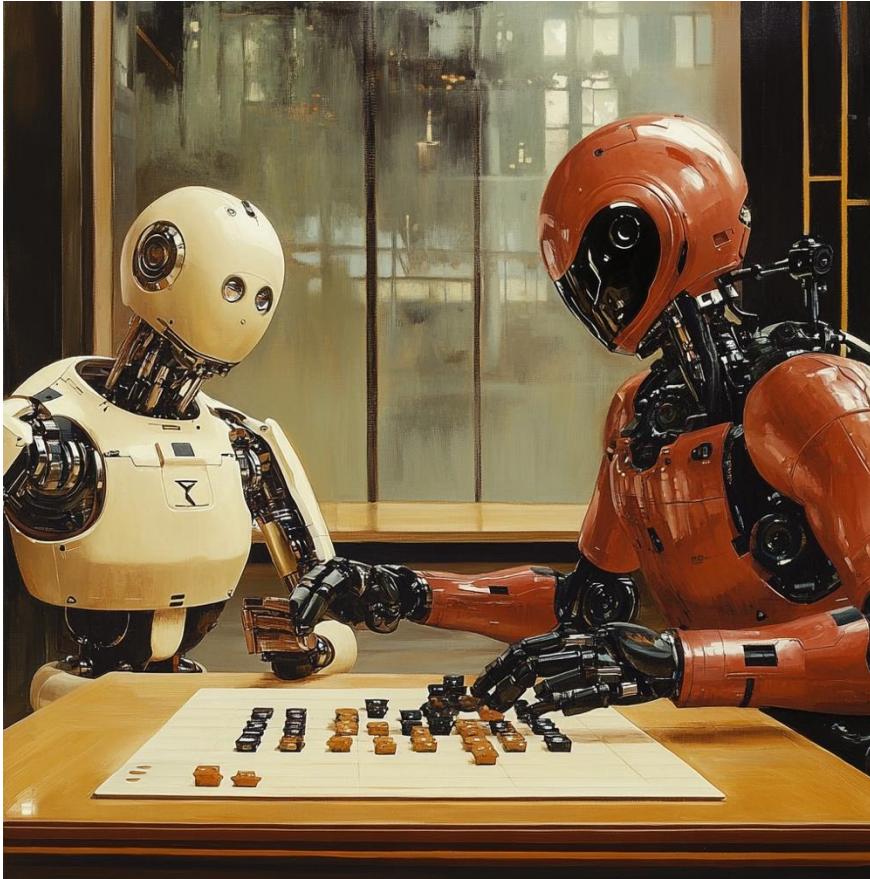
---

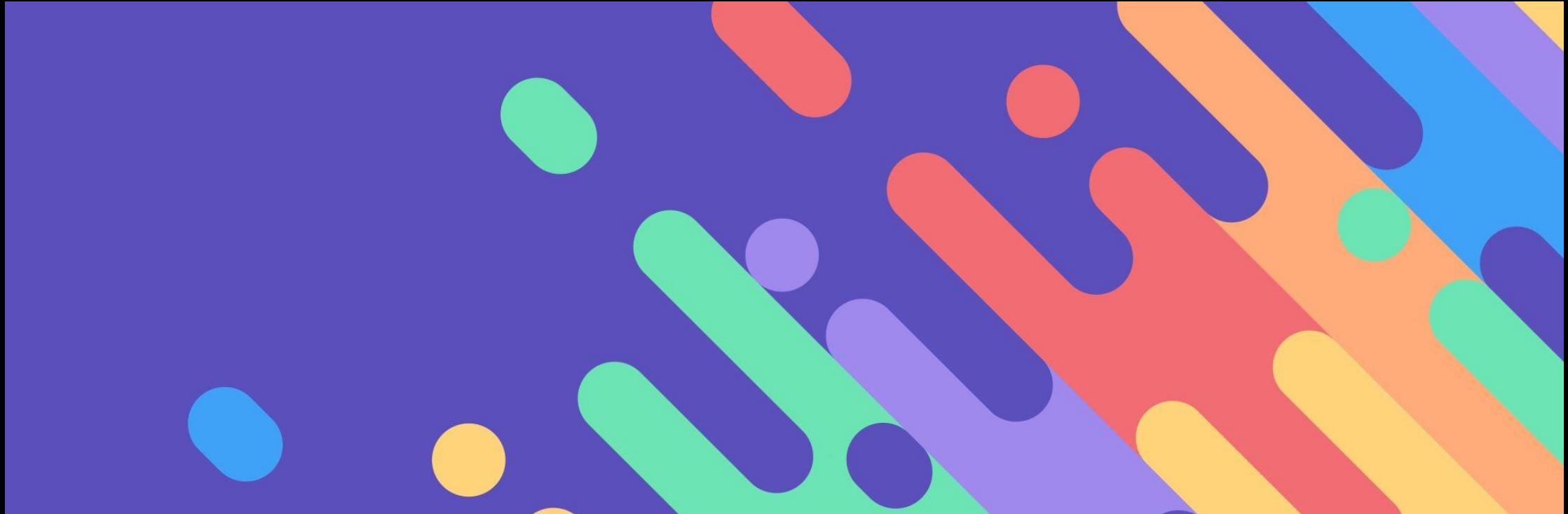
# APRENDIZAJE POR REFUERZO



---

# APRENDIZAJE POR REFUERZO





---

# PROCESO DE DECISIÓN DE MÁRKOV

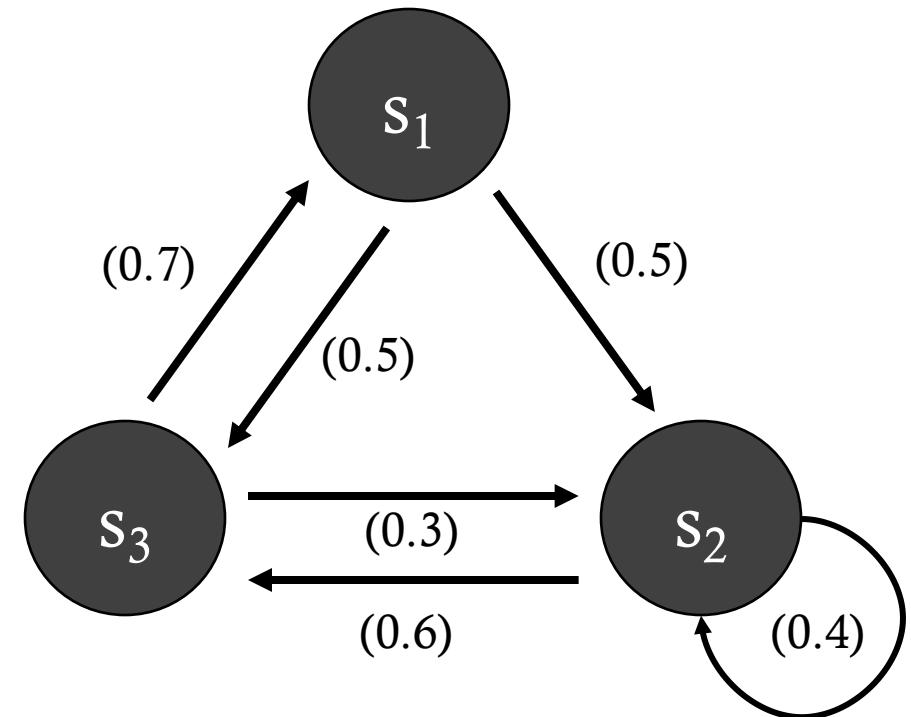
# PROCESO DE DECISIÓN DE MÁRKOV

Son una extensión de las cadenas de Márkov.

Pero primero, ¿qué es una cadena de Márkov?

*Una cadena de Márkov es un proceso estocástico en el que el siguiente estado depende sólo del estado actual, no del camino previo.*

$$P(s_{t+1} | s_t, s_{t-1}, \dots, s_0) = P(s_{t+1} | s_t)$$

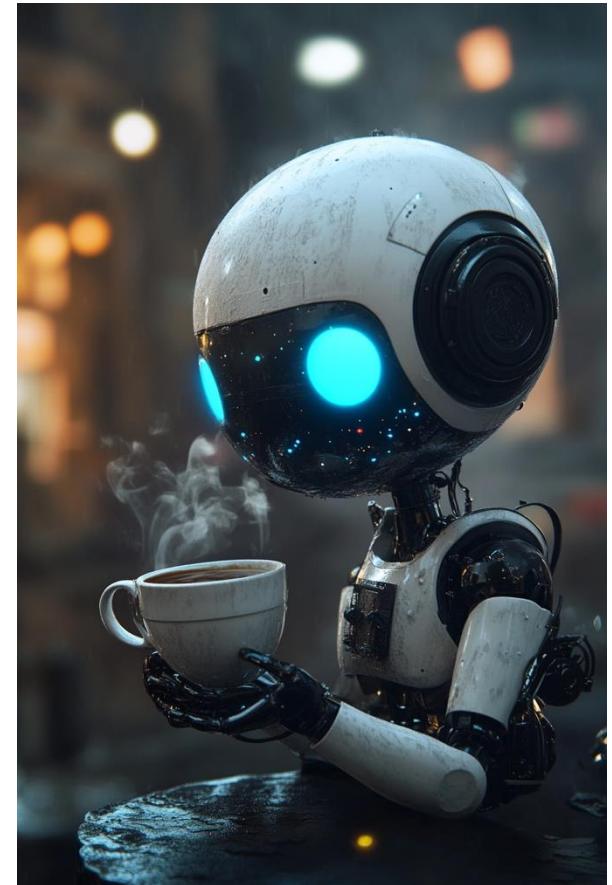


---

# PROCESO DE DECISIÓN DE MÁRKOV

Un ejemplo:

Podemos construir un *chatbot* que, a partir de muchas conversaciones previas, genere texto automáticamente comenzando con una frase inicial. Esto se logra modelando las transiciones entre palabras o frases como una cadena de Márkov.

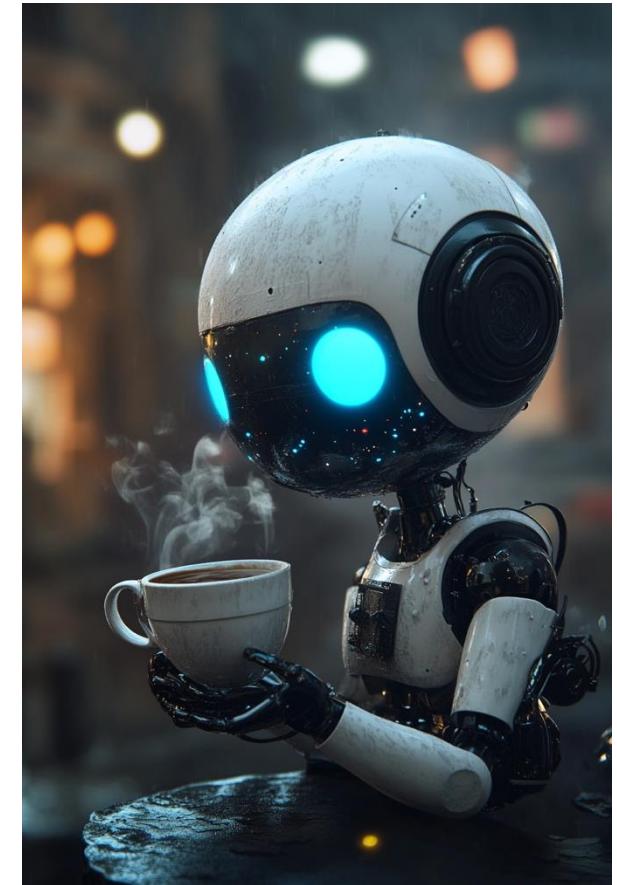
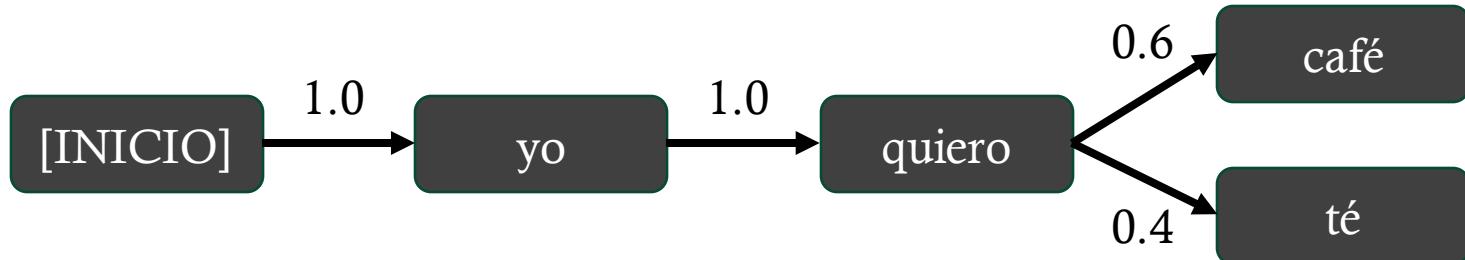


# PROCESO DE DECISIÓN DE MÁRKOV

Un ejemplo:

Podemos construir un *chatbot* que, a partir de muchas conversaciones previas, genere texto automáticamente comenzando con una frase inicial. Esto se logra modelando las transiciones entre palabras o frases como una cadena de Márkov.

Así obtenemos secuencias como:

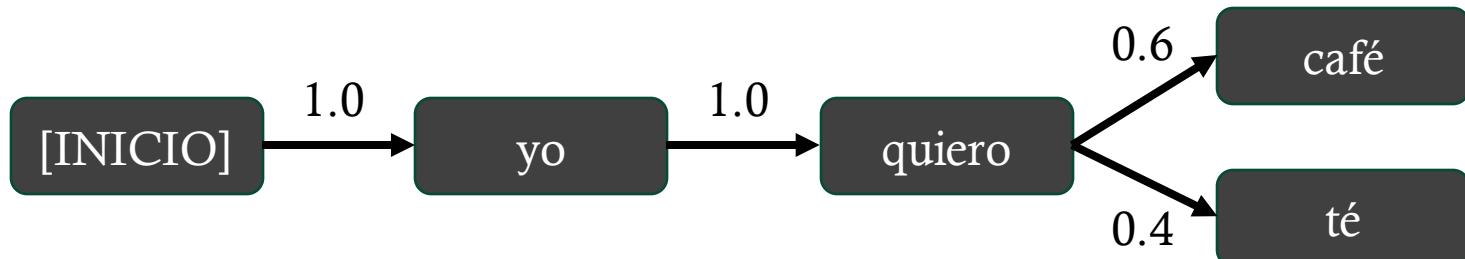


# PROCESO DE DECISIÓN DE MÁRKOV

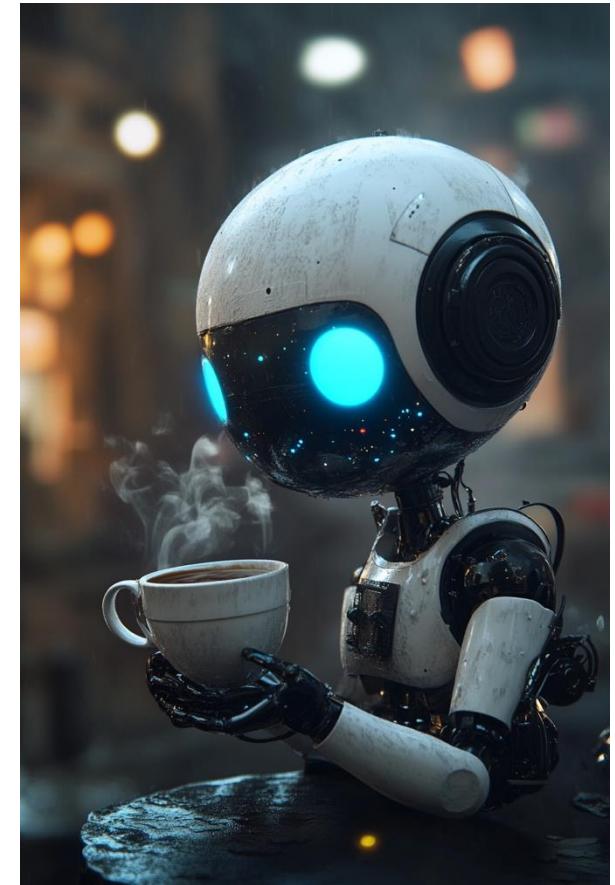
Un ejemplo:

Podemos construir un *chatbot* que, a partir de muchas conversaciones previas, genere texto automáticamente comenzando con una frase inicial. Esto se logra modelando las transiciones entre palabras o frases como una cadena de Márkov.

Así obtenemos secuencias como:



Chatbot:

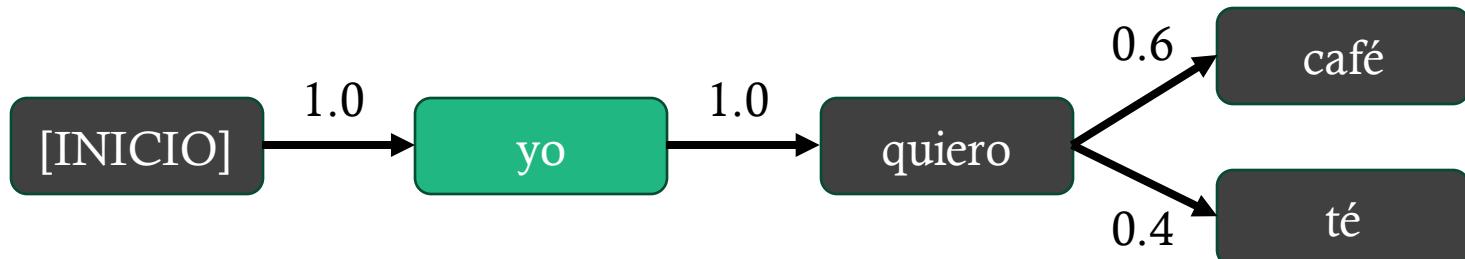


# PROCESO DE DECISIÓN DE MÁRKOV

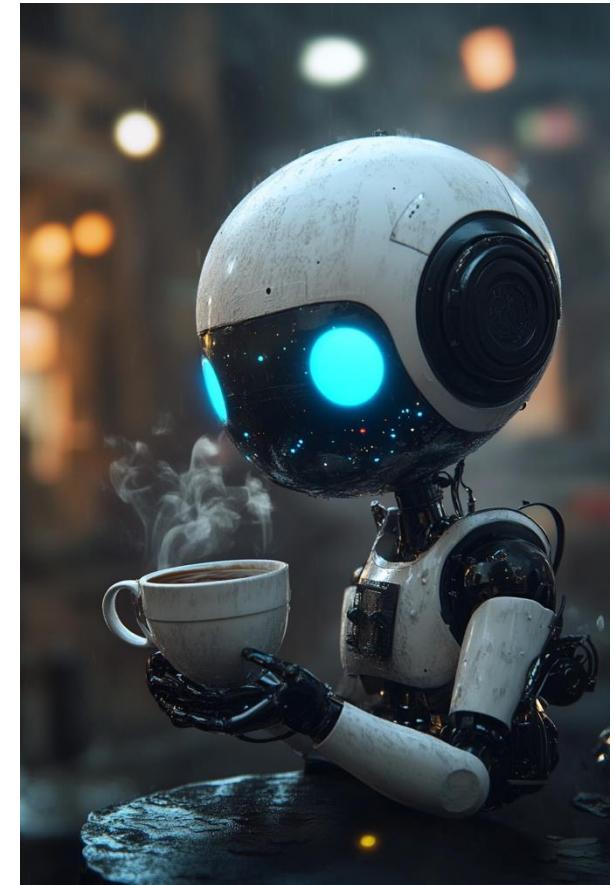
Un ejemplo:

Podemos construir un *chatbot* que, a partir de muchas conversaciones previas, genere texto automáticamente comenzando con una frase inicial. Esto se logra modelando las transiciones entre palabras o frases como una cadena de Márkov.

Así obtenemos secuencias como:



Chatbot: Yo

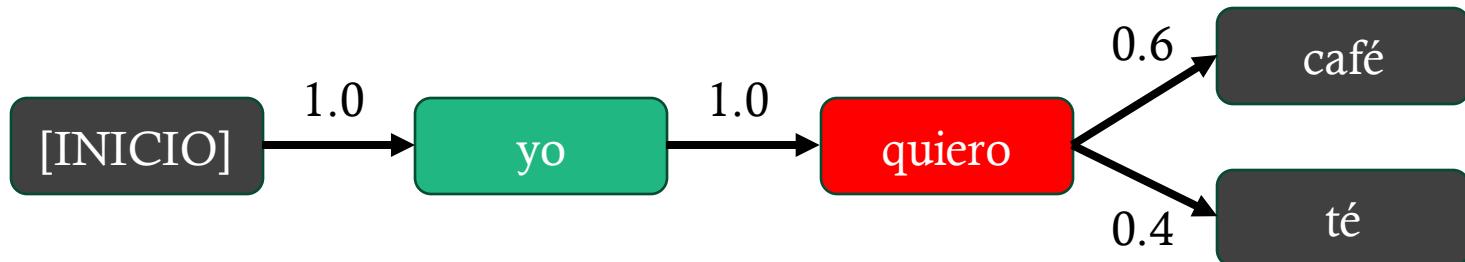


# PROCESO DE DECISIÓN DE MÁRKOV

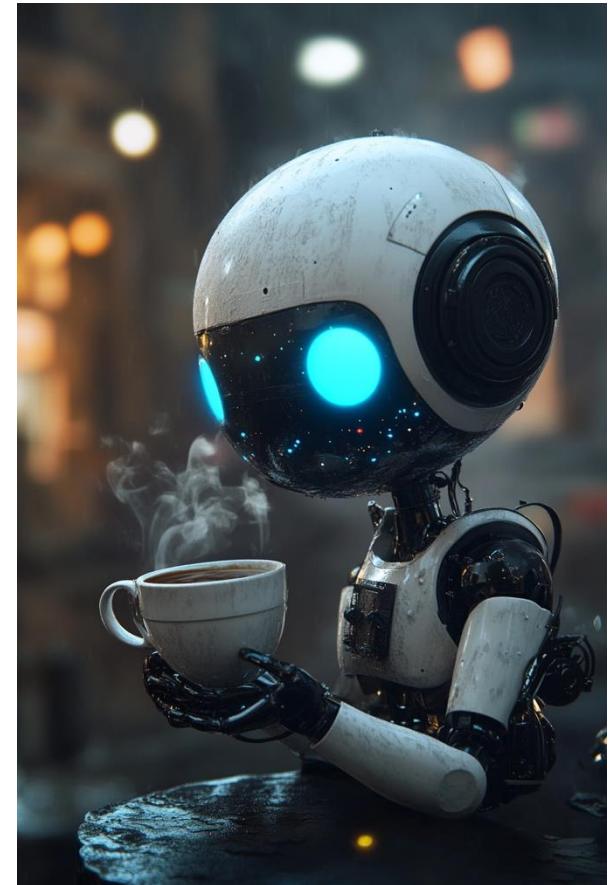
Un ejemplo:

Podemos construir un *chatbot* que, a partir de muchas conversaciones previas, genere texto automáticamente comenzando con una frase inicial. Esto se logra modelando las transiciones entre palabras o frases como una cadena de Márkov.

Así obtenemos secuencias como:



Chatbot: Yo quiero

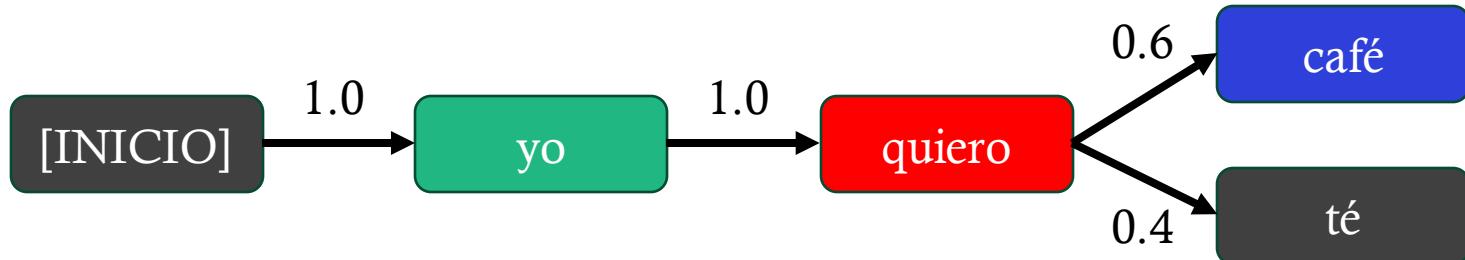


# PROCESO DE DECISIÓN DE MÁRKOV

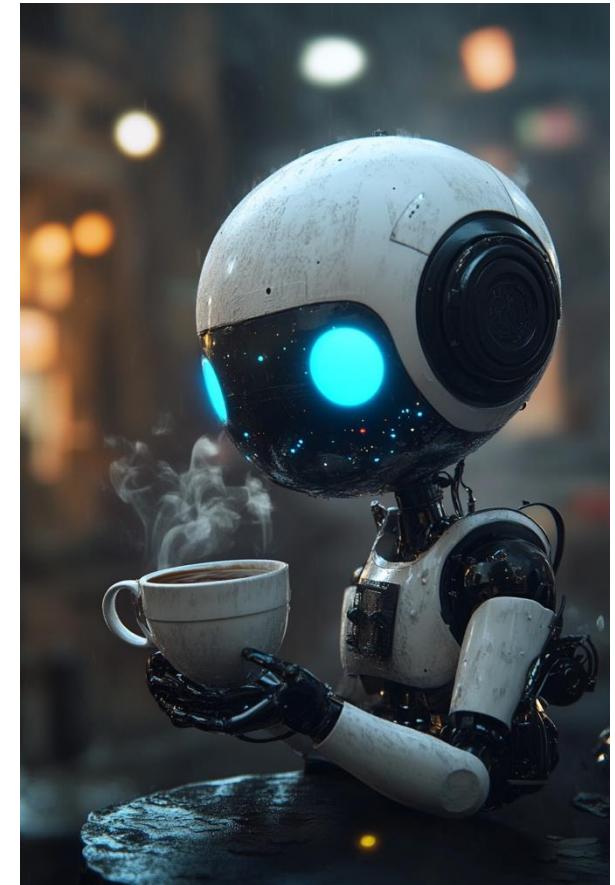
Un ejemplo:

Podemos construir un *chatbot* que, a partir de muchas conversaciones previas, genere texto automáticamente comenzando con una frase inicial. Esto se logra modelando las transiciones entre palabras o frases como una cadena de Márkov.

Así obtenemos secuencias como:



Chatbot: Yo quiero café



---

# PROCESO DE DECISIÓN DE MÁRKOV

\* Lo que ya sabemos:

- En una cadena de Márkov hay estados y probabilidades de transición entre ellos.
- No hay control: el sistema evoluciona automáticamente.

⌚ ¿Qué agregamos en un Proceso de Decisión de Márkov (MDP)?

- **Agente** → Toma decisiones en cada estado.
- **Acciones** → Elige cómo actuar.
- **Recompensa** → Cada acción tiene una consecuencia (positiva o negativa).
- **Política** → Define qué acción tomar en cada estado.
- **Objetivo** → Maximizar la recompensa acumulada a largo plazo.

Ahora, la probabilidad de transición depende también de la acción tomada:

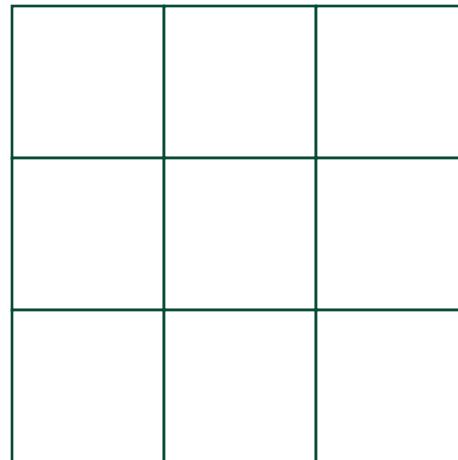
$$P(s_{t+1} | s_t, a_n)$$

---

# PROCESO DE DECISIÓN DE MÁRKOV



Entorno

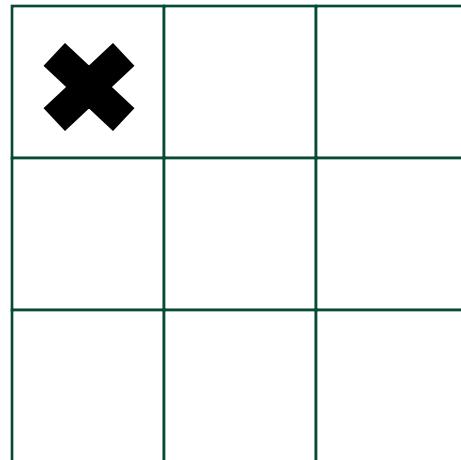


Agente

# PROCESO DE DECISIÓN DE MÁRKOV



Entorno

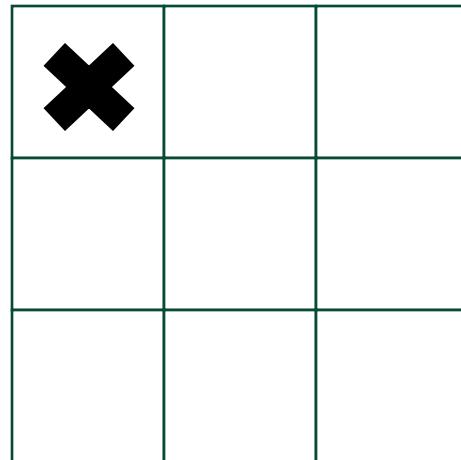


Agente

# PROCESO DE DECISIÓN DE MÁRKOV



Entorno

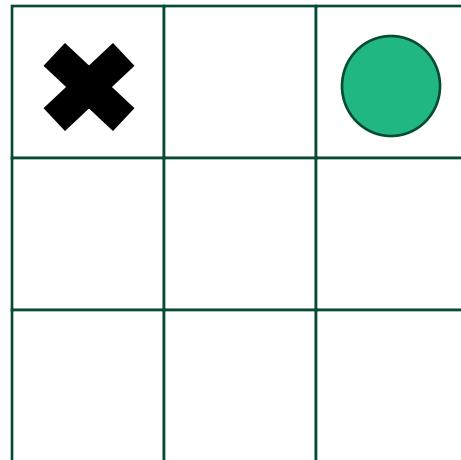


Agente

# PROCESO DE DECISIÓN DE MÁRKOV



Entorno

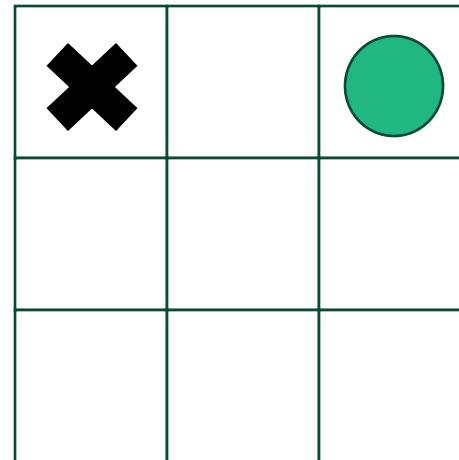


Agente

# PROCESO DE DECISIÓN DE MÁRKOV

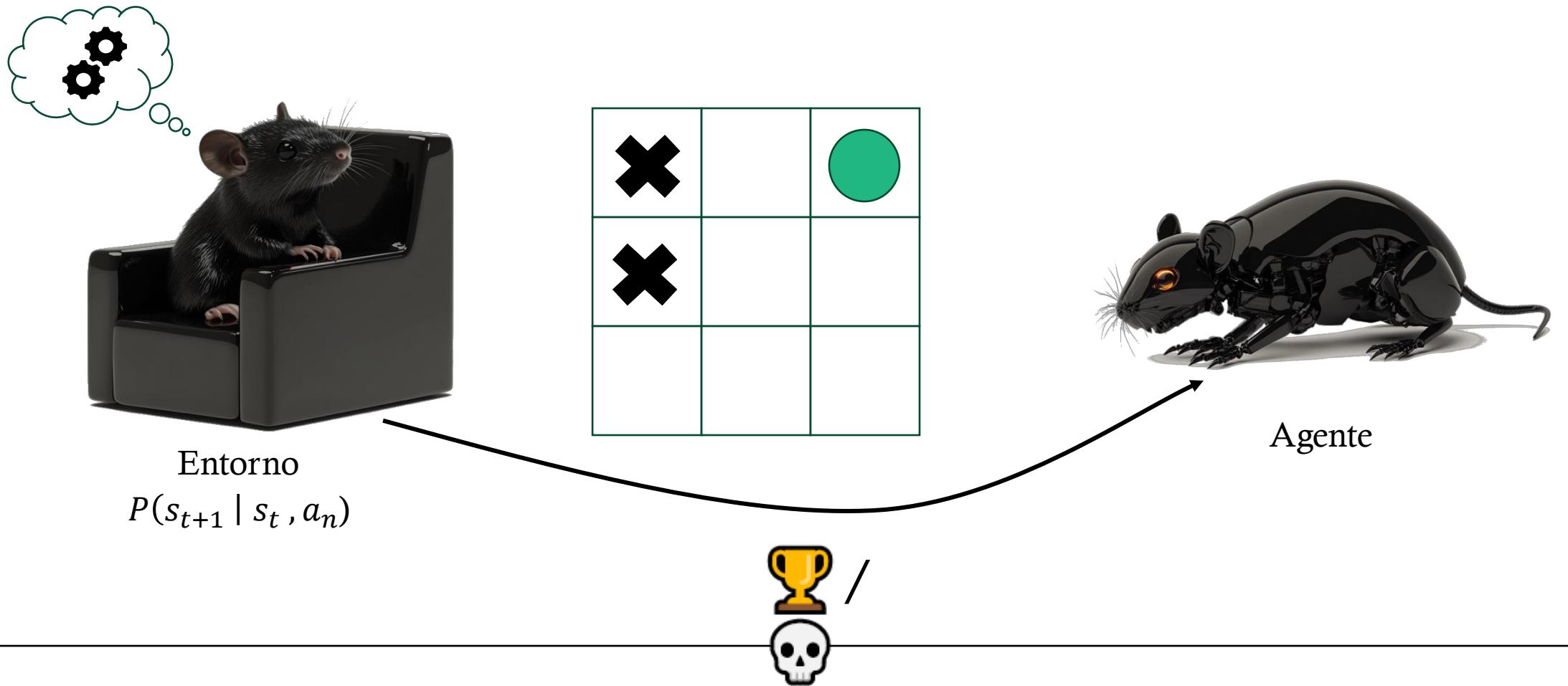


Entorno



Agente

# PROCESO DE DECISIÓN DE MÁRKOV

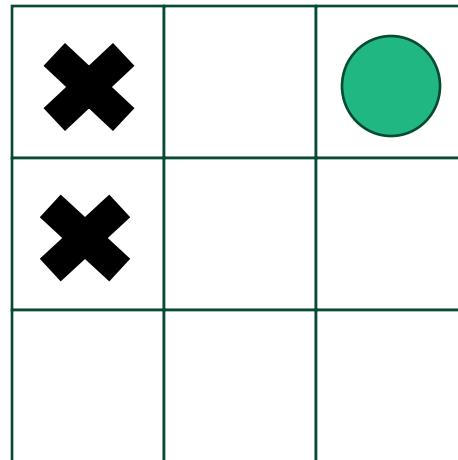


---

# PROCESO DE DECISIÓN DE MÁRKOV



Entorno



Agente

---

# PROCESO DE DECISIÓN DE MÁRKOV

La ejecución de un MDP puede describirse como una secuencia de estados, acciones y recompensas a lo largo del tiempo.

- En tareas con un final definido, esta secuencia se denomina episódica: **cada episodio es independiente del siguiente**, y el ciclo de aprendizaje por refuerzo (AR) se repite en cada episodio, compuesto por múltiples pasos de tiempo.
- Por otro lado, cuando la tarea no tiene un final determinado, se trata de una **tarea continua**, que puede prolongarse indefinidamente.

---

# PROCESO DE DECISIÓN DE MÁRKOV

Interacción entre el **agente** y el **entorno**.

El MDP funciona alternando entre decisiones del **agente** y respuestas del **entorno**. En cada paso de tiempo:



El **agente** elige una acción a partir del estado actual.



El **entorno**, dada esa acción y el estado actual, determina el siguiente estado y la recompensa asociada.

---

# PROCESO DE DECISIÓN DE MÁRKOV

¿Cómo toma decisiones el agente?

Este es el núcleo del problema de aprendizaje por refuerzo. Para resolverlo, se usan tres conceptos clave:

- **Retorno**: la suma de las recompensas acumuladas a lo largo del tiempo.
- **Política**: la estrategia que define qué acción tomar en cada estado.
- **Valor**: el retorno esperado al seguir una política determinada desde un estado.

---

# PROCESO DE DECISIÓN DE MÁRKOV

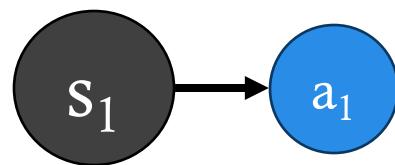
¿Cómo toma decisiones el agente?



---

# PROCESO DE DECISIÓN DE MÁRKOV

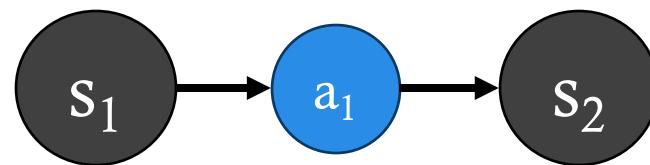
¿Cómo toma decisiones el agente?



---

# PROCESO DE DECISIÓN DE MÁRKOV

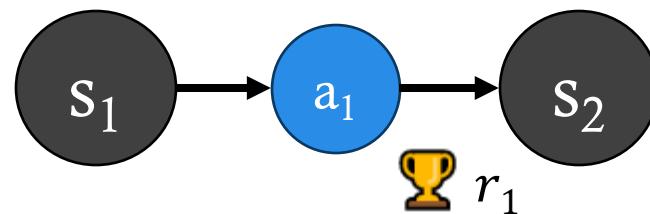
¿Cómo toma decisiones el agente?



---

# PROCESO DE DECISIÓN DE MÁRKOV

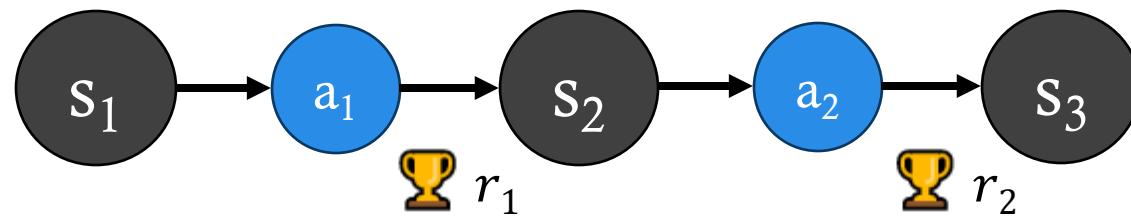
¿Cómo toma decisiones el agente?



---

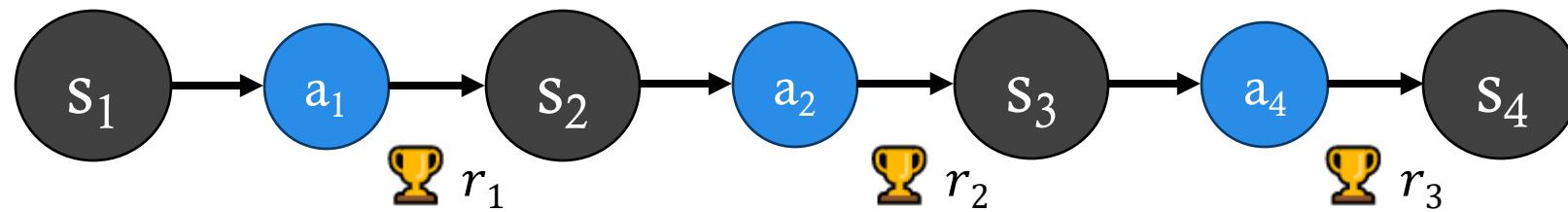
# PROCESO DE DECISIÓN DE MÁRKOV

¿Cómo toma decisiones el agente?



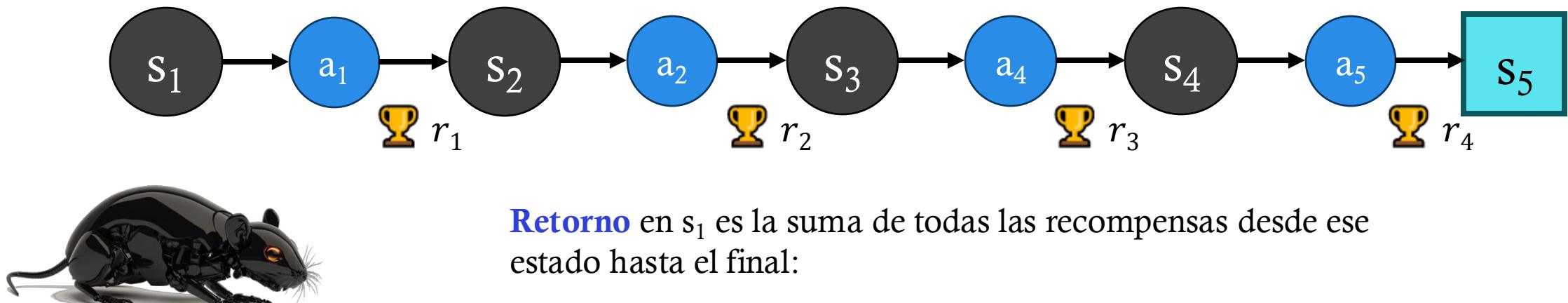
# PROCESO DE DECISIÓN DE MÁRKOV

¿Cómo toma decisiones el agente?



# PROCESO DE DECISIÓN DE MÁRKOV

¿Cómo toma decisiones el agente?



$$\text{Retorno}_{s_1} = r_1 + r_2 + r_3 + r_4$$

---

# PROCESO DE DECISIÓN DE MÁRKOV

Retorno con descuento

Al calcular el retorno, no se suman simplemente todas las recompensas. Se aplica **un factor de descuento  $\gamma$**  que reduce el valor de las recompensas futuras:

$$\text{Retorno} = r_0 + \gamma r_1 + \gamma^2 r_2 + \cdots + \gamma^n r_n$$

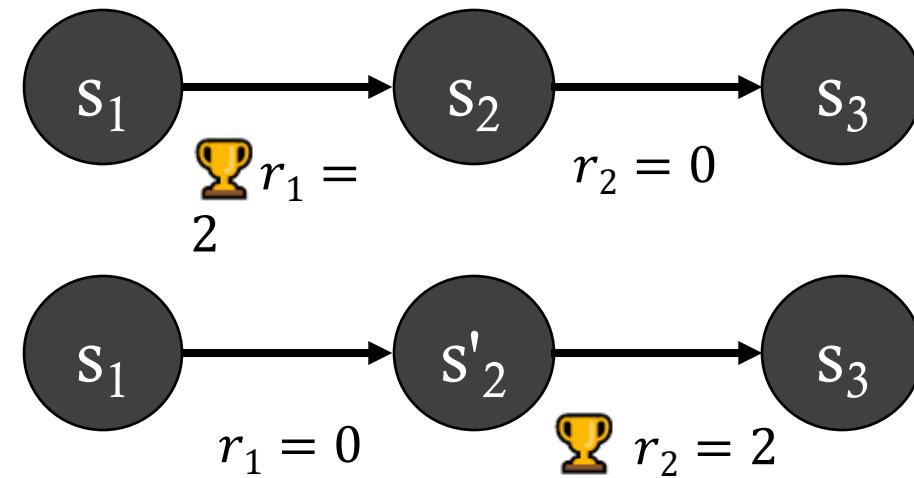
Esto **evita que el retorno crezca indefinidamente** en tareas de muchos pasos, y valora más las recompensas inmediatas.

---

# PROCESO DE DECISIÓN DE MÁRKOV

Recompensas inmediatas vs. futuras

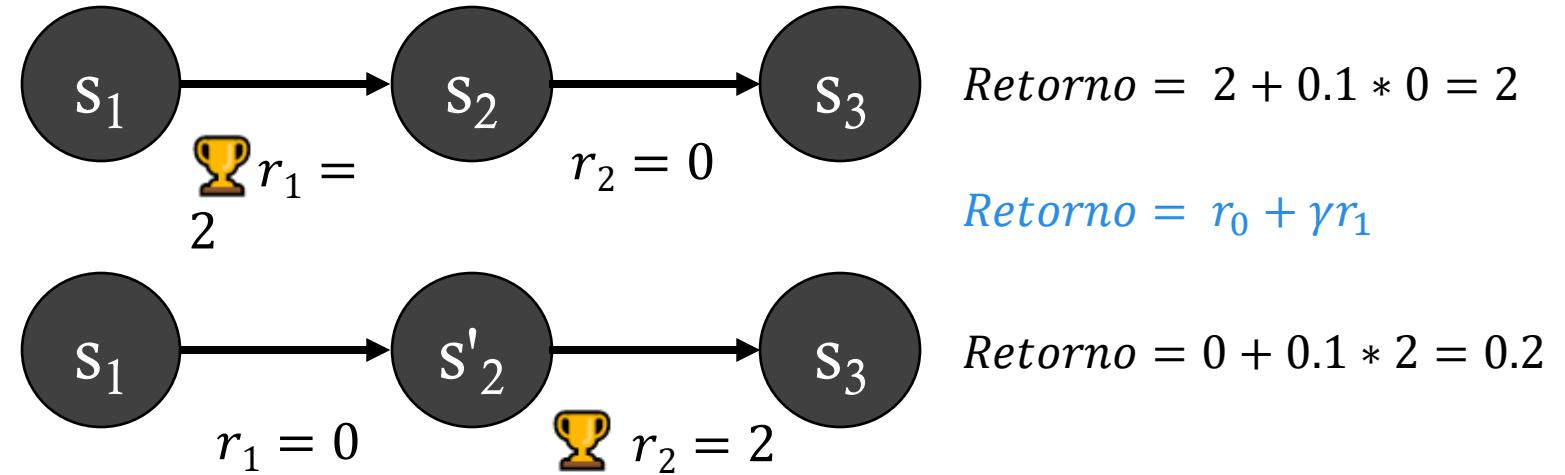
La recompensa inmediata suele ser más valiosa que una recompensa futura. Por eso se usa el factor de descuento: para reflejar esta preferencia en el cálculo del retorno.



# PROCESO DE DECISIÓN DE MÁRKOV

Recompensas inmediatas vs. futuras

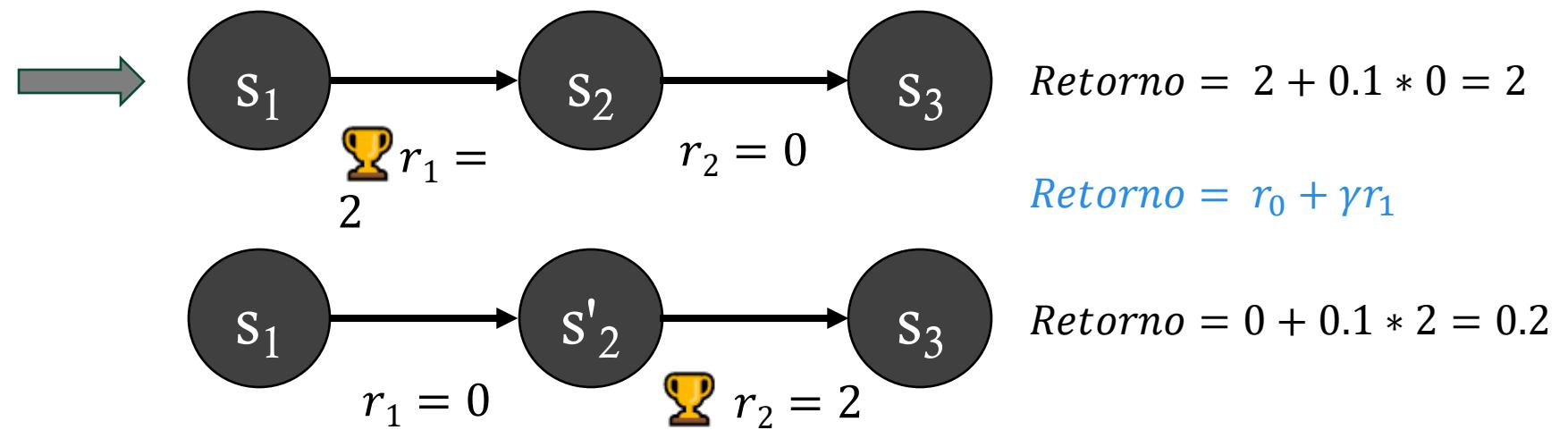
La recompensa inmediata suele ser más valiosa que una recompensa futura. Por eso se usa el factor de descuento: para reflejar esta preferencia en el cálculo del retorno.



# PROCESO DE DECISIÓN DE MÁRKOV

Recompensas inmediatas vs. futuras

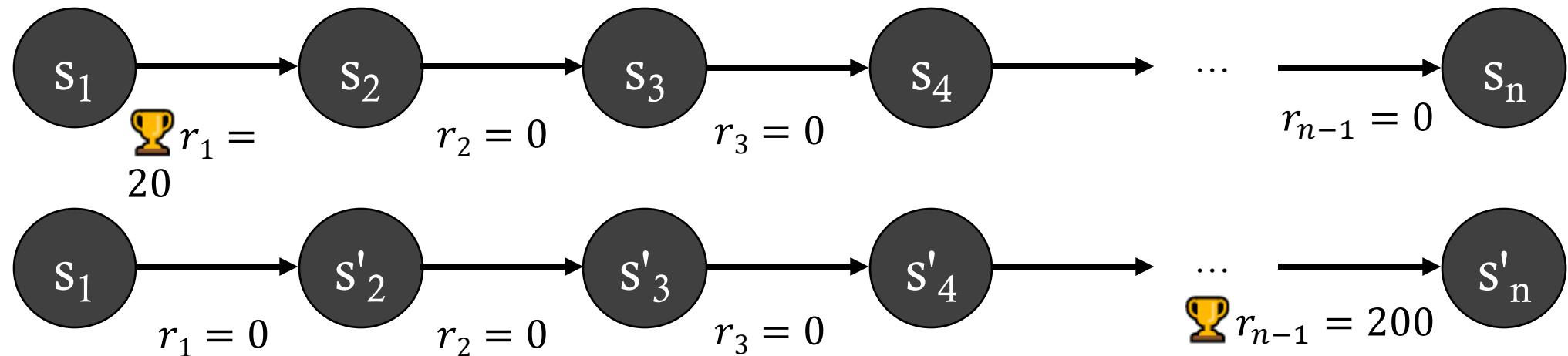
La recompensa inmediata suele ser más valiosa que una recompensa futura. Por eso se usa el factor de descuento: para reflejar esta preferencia en el cálculo del retorno.



# PROCESO DE DECISIÓN DE MÁRKOV

Decisiones con recompensa futura

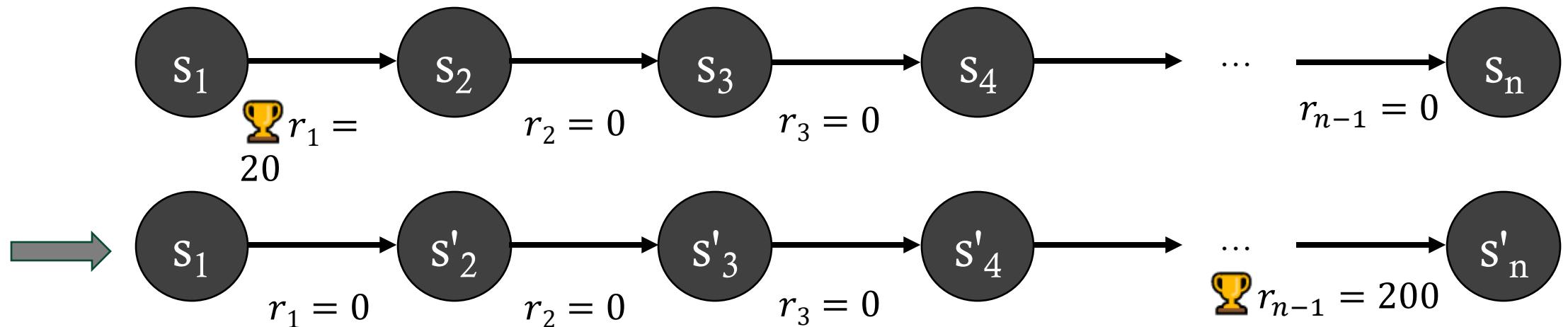
Aun así, si una recompensa futura es lo suficientemente grande, puede ser preferible esperar. El agente debe aprender a equilibrar recompensas inmediatas y futuras para maximizar su retorno.



# PROCESO DE DECISIÓN DE MÁRKOV

Decisiones con recompensa futura

Aun así, si una recompensa futura es lo suficientemente grande, puede ser preferible esperar. El agente debe aprender a equilibrar recompensas inmediatas y futuras para maximizar su retorno.



---

# PROCESO DE DECISIÓN DE MÁRKOV

## Política

La política define qué acción tomar en cada estado.

Una forma sencilla de representarla es mediante una tabla estado-acción, donde se indica la probabilidad de elegir cada acción según el estado.

	a1	a2	a3
s1	$p(a_1 s_1)$	...	...
s2	...	...	...
s3	...	...	...
s4	...	...	$p(a_3 s_4)$

# PROCESO DE DECISIÓN DE MÁRKOV

## Política

La política define qué acción tomar en cada estado.

Una forma sencilla de representarla es mediante una tabla estado-acción, donde se indica la probabilidad de elegir cada acción según el estado.

Política	$a_1$	$a_2$	$a_3$
$s_1$	$p(a_1 s_1)$	...	...
$s_2$	...	...	...
$s_3$	...	...	...
$s_4$	...	...	$p(a_3 s_4)$

Probabilidad  
de realizar la  
acción  $a_3$   
cuando se  
encuentra en  
el estado  $s_4$

---

# PROCESO DE DECISIÓN DE MÁRKOV

Ejemplos de política

Una política puede representarse con una tabla como esta:

Política	$a_1$	$a_2$	$a_3$
$s_1$	0.5	0.3	0.2
$s_2$	0.1	0.1	0.8
$s_3$	0.6	0.4	--
$s_4$	0.2	0.4	0.4

# PROCESO DE DECISIÓN DE MÁRKOV

Ejemplos de política

Política	a <sub>1</sub>	a <sub>2</sub>	a <sub>3</sub>
s <sub>1</sub>	0.5	0.3	0.2
s <sub>2</sub>	0.1	0.1	0.8
s <sub>3</sub>	0.6	0.4	--
s <sub>4</sub>	0.2	0.4	0.4

Política estocástica

Política	a <sub>1</sub>	a <sub>2</sub>	a <sub>3</sub>
s <sub>1</sub>	1	0	0
s <sub>2</sub>	0	0	1
s <sub>3</sub>	1	0	--
s <sub>4</sub>	0	1	0

Política determinística

# PROCESO DE DECISIÓN DE MÁRKOV

## Política

Pol	a <sub>1</sub>	a <sub>2</sub>	a <sub>3</sub>
s <sub>1</sub>	1/3	1/3	1/3
s <sub>2</sub>	1/3	1/3	1/3
s <sub>3</sub>	1/2	1/2	--
s <sub>4</sub>	1/3	1/3	1/3



Pol	a <sub>1</sub>	a <sub>2</sub>	a <sub>3</sub>
s <sub>1</sub>	0.5	0.3	0.2
s <sub>2</sub>	0.1	0.1	0.8
s <sub>3</sub>	0.6	0.4	--
s <sub>4</sub>	0.2	0.4	0.4

# PROCESO DE DECISIÓN DE MÁRKOV

Objetivo de la política

Pol	a <sub>1</sub>	a <sub>2</sub>	a <sub>3</sub>
s <sub>1</sub>	1/3	1/3	1/3
s <sub>2</sub>	1/3	1/3	1/3
s <sub>3</sub>	1/2	1/2	--
s <sub>4</sub>	1/3	1/3	1/3



Pol	a <sub>1</sub>	a <sub>2</sub>	a <sub>3</sub>
s <sub>1</sub>	0.5	0.3	0.2
s <sub>2</sub>	0.1	0.1	0.8
s <sub>3</sub>	0.6	0.4	--
s <sub>4</sub>	0.2	0.4	0.4

*El objetivo del agente es seguir una política que maximice su retorno acumulado a largo plazo.*

---

# PROCESO DE DECISIÓN DE MÁRKOV

¿Qué significa "valor"?

Si un agente parte de un estado y siempre actúa siguiendo una política determinada...

¿Cuál es el rendimiento que puede esperar obtener?

Política	a <sub>1</sub>	a <sub>2</sub>	a <sub>3</sub>
s <sub>1</sub>	0.5	0.3	0.2
s <sub>2</sub>	0.1	0.1	0.8
s <sub>3</sub>	0.6	0.4	--
s <sub>4</sub>	0.2	0.4	0.4



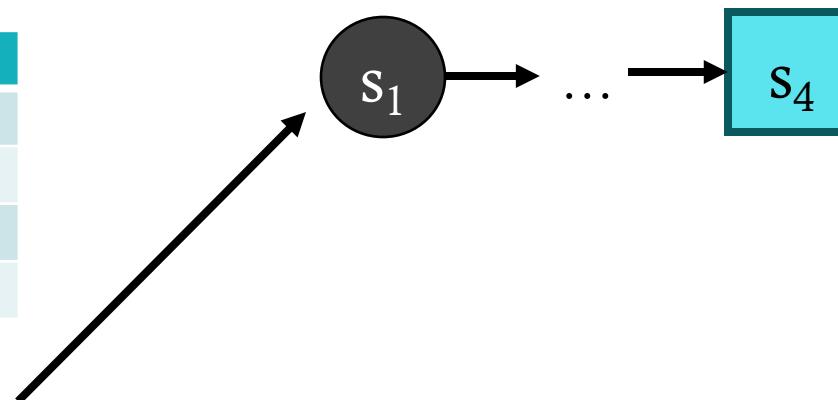
# PROCESO DE DECISIÓN DE MÁRKOV

¿Qué significa "valor"?

Si un agente parte de un estado y siempre actúa siguiendo una política determinada...

¿Cuál es el rendimiento que puede esperar obtener?

Política	a <sub>1</sub>	a <sub>2</sub>	a <sub>3</sub>
s <sub>1</sub>	0.5	0.3	0.2
s <sub>2</sub>	0.1	0.1	0.8
s <sub>3</sub>	0.6	0.4	--
s <sub>4</sub>	0.2	0.4	0.4



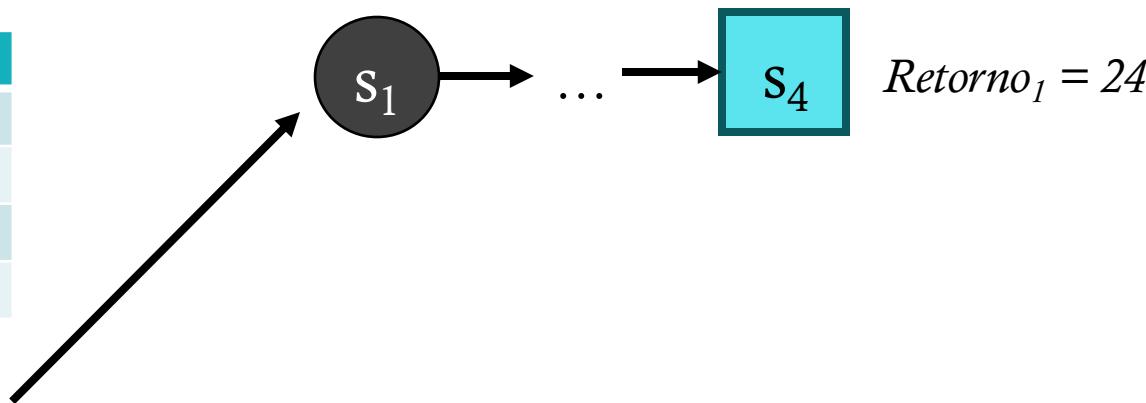
# PROCESO DE DECISIÓN DE MÁRKOV

¿Qué significa "valor"?

Si un agente parte de un estado y siempre actúa siguiendo una política determinada...

¿Cuál es el rendimiento que puede esperar obtener?

Política	a <sub>1</sub>	a <sub>2</sub>	a <sub>3</sub>
s <sub>1</sub>	0.5	0.3	0.2
s <sub>2</sub>	0.1	0.1	0.8
s <sub>3</sub>	0.6	0.4	--
s <sub>4</sub>	0.2	0.4	0.4



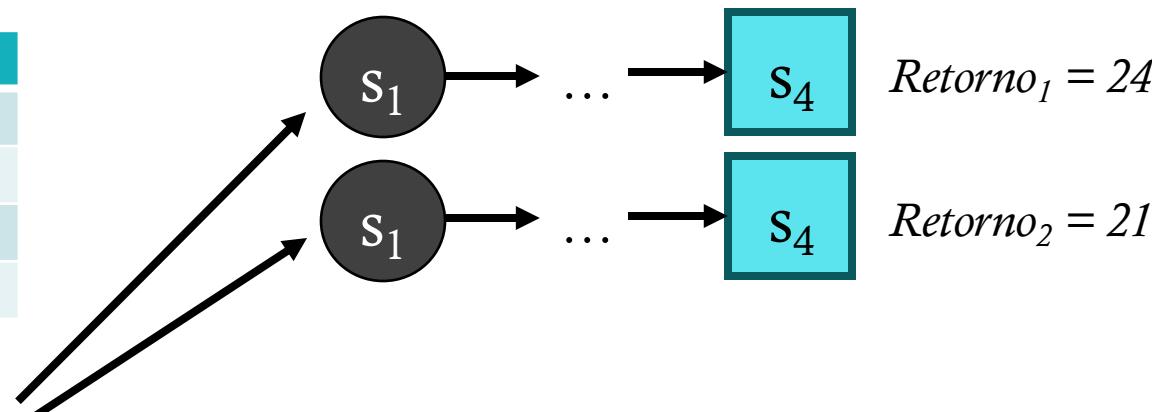
# PROCESO DE DECISIÓN DE MÁRKOV

¿Qué significa "valor"?

Si un agente parte de un estado y siempre actúa siguiendo una política determinada...

¿Cuál es el rendimiento que puede esperar obtener?

Política	a <sub>1</sub>	a <sub>2</sub>	a <sub>3</sub>
s <sub>1</sub>	0.5	0.3	0.2
s <sub>2</sub>	0.1	0.1	0.8
s <sub>3</sub>	0.6	0.4	--
s <sub>4</sub>	0.2	0.4	0.4



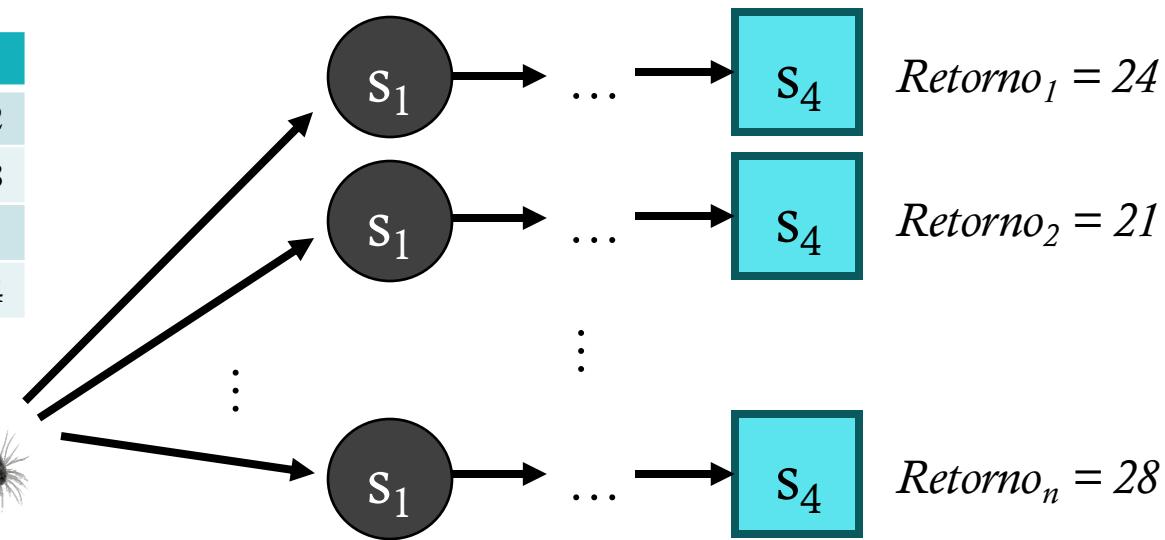
# PROCESO DE DECISIÓN DE MÁRKOV

¿Qué significa "valor"?

Si un agente parte de un estado y siempre actúa siguiendo una política determinada...

¿Cuál es el rendimiento que puede esperar obtener?

Política	a <sub>1</sub>	a <sub>2</sub>	a <sub>3</sub>
s <sub>1</sub>	0.5	0.3	0.2
s <sub>2</sub>	0.1	0.1	0.8
s <sub>3</sub>	0.6	0.4	--
s <sub>4</sub>	0.2	0.4	0.4



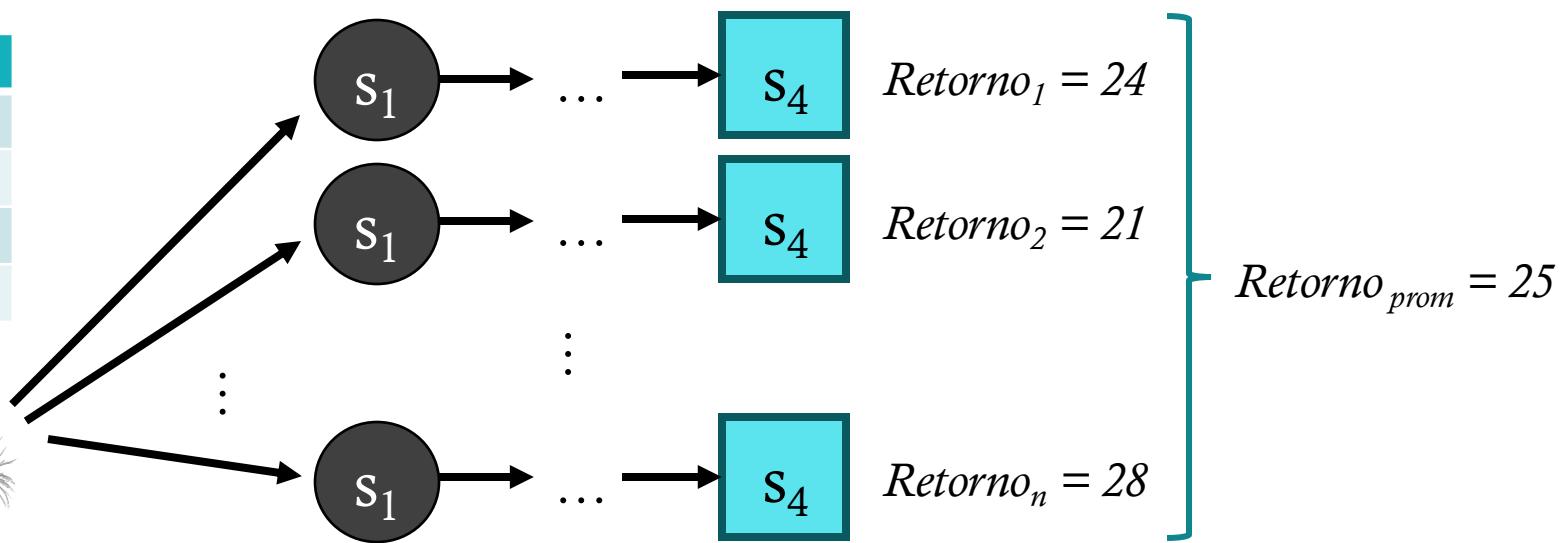
# PROCESO DE DECISIÓN DE MÁRKOV

¿Qué significa "valor"?

Si un agente parte de un estado y siempre actúa siguiendo una política determinada...

¿Cuál es el rendimiento que puede esperar obtener?

Política	a <sub>1</sub>	a <sub>2</sub>	a <sub>3</sub>
s <sub>1</sub>	0.5	0.3	0.2
s <sub>2</sub>	0.1	0.1	0.8
s <sub>3</sub>	0.6	0.4	--
s <sub>4</sub>	0.2	0.4	0.4

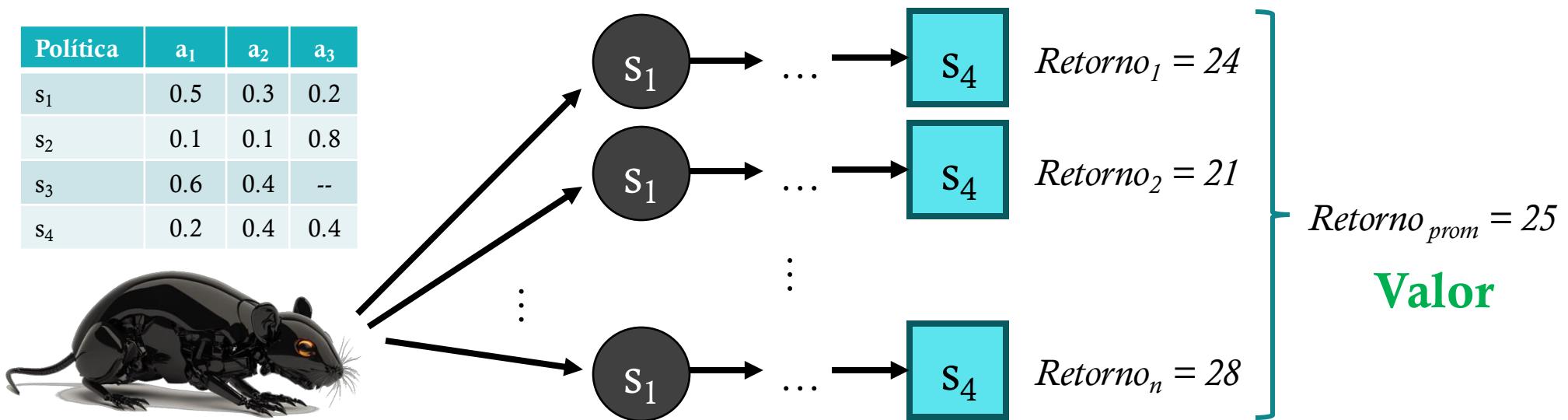


# PROCESO DE DECISIÓN DE MÁRKOV

¿Qué significa "valor"?

Si un agente parte de un estado y siempre actúa siguiendo una política determinada...

¿Cuál es el rendimiento que puede esperar obtener?



---

# PROCESO DE DECISIÓN DE MÁRKOV

## Tipos de Valor

- **Valor de estado  $V_\pi(s)$ :** retorno esperado al comenzar desde un **estado s** y seguir una **política  $\pi$** .
- **Valor de estado-acción  $Q_\pi(s, a)$ :** retorno esperado al realizar una **acción a** en el **estado s** y siguiendo una **política  $\pi$** .

# PROCESO DE DECISIÓN DE MÁRKOV

## Política y valor

Un agente puede almacenar conocimiento usando la tabla de política o también la tabla de valores, las cuales ambos son equivalentes:

Política	a <sub>1</sub>	a <sub>2</sub>	a <sub>3</sub>
s <sub>1</sub>	0.5	0.3	0.2
s <sub>2</sub>	0.1	0.1	0.8
s <sub>3</sub>	0.6	0.4	--
s <sub>4</sub>	0.2	0.4	0.4



Q	a <sub>1</sub>	a <sub>2</sub>	a <sub>3</sub>
s <sub>1</sub>	50	30	20
s <sub>2</sub>	13	13	106
s <sub>3</sub>	86	58	--
s <sub>4</sub>	8	16	16

---

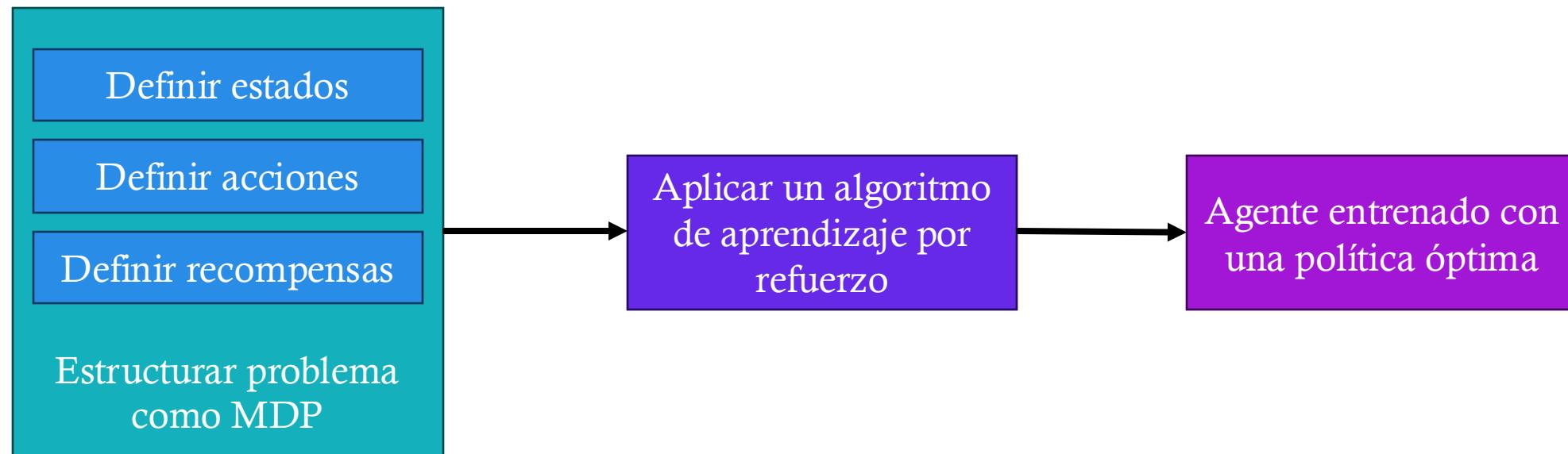
# PROCESO DE DECISIÓN DE MÁRKOV

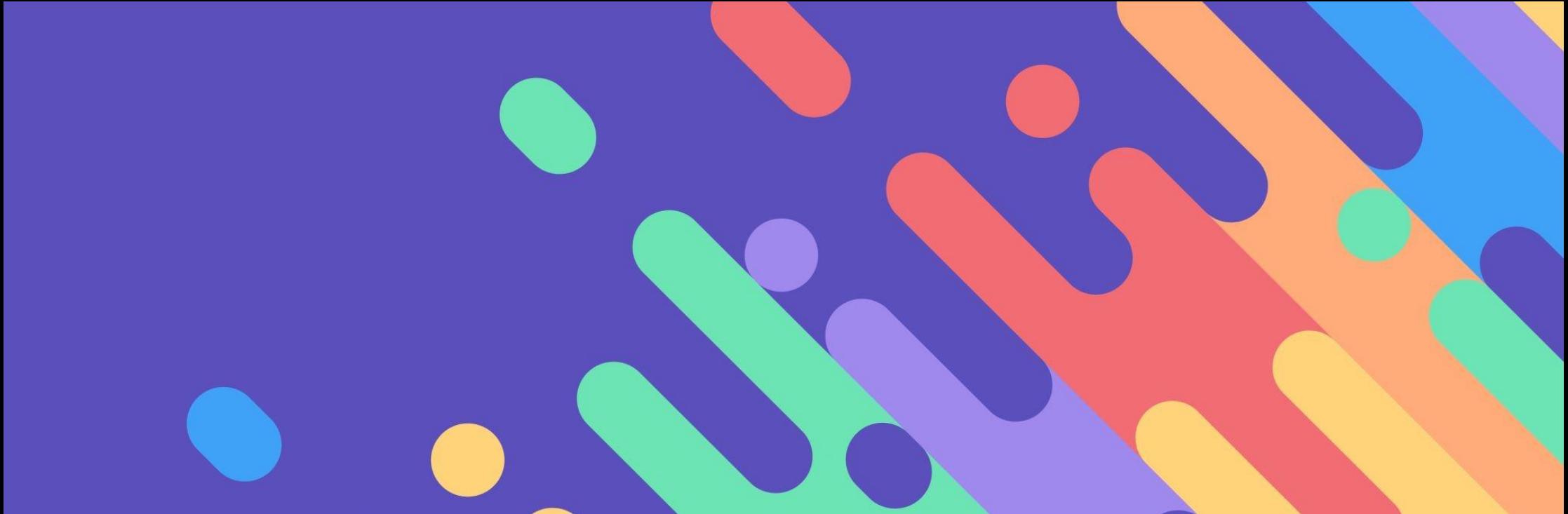
Resolviendo el problema del aprendizaje por refuerzo: encontrar la política óptima

- Podemos usar **la función de valor** para comparar políticas y determinar cuáles son **buenas** y **cuáles no**.
- También se puede utilizar para encontrar la **mejor** política posible, conocida como **política óptima**.
- Por lo tanto, entrenar un agente mediante **aprendizaje por refuerzo** y **MDP** implica buscar la **política óptima** para maximizar el retorno del agente.

# PROCESO DE DECISIÓN DE MÁRKOV

Resolviendo el problema del aprendizaje por refuerzo: encontrar la política óptima





---

# CATEGORÍAS DE SOLUCIONES

---

# CATEGORÍAS DE SOLUCIONES

Existen muchos algoritmos para encontrar la **política óptima** de un agente. Estos se agrupan en dos grandes categorías:

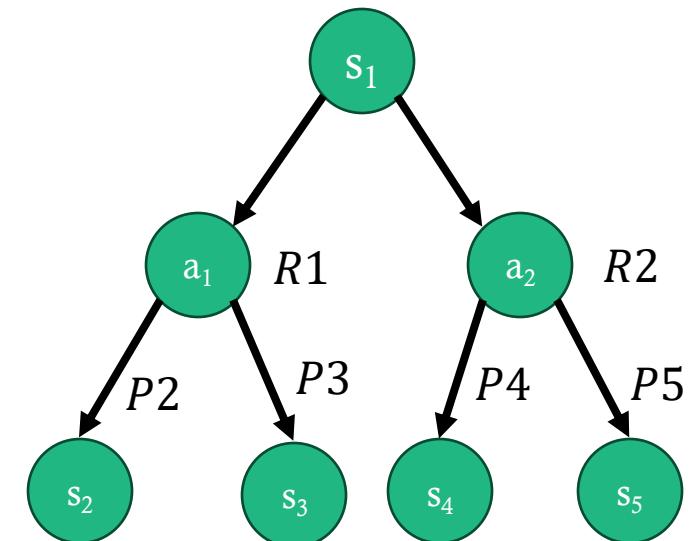
- Basado en modelo
- Libres de modelo (o sin modelo)

# CATEGORÍAS DE SOLUCIONES

Basado en modelo

Los enfoques basados en modelo se utilizan cuando se conoce el funcionamiento interno del entorno. *Es decir, se puede predecir con certeza el siguiente estado y la recompensa que resultarán de ejecutar una acción desde un estado dado.*

Esto permite al agente planificar, de forma similar a los algoritmos de búsqueda que ya conocemos.

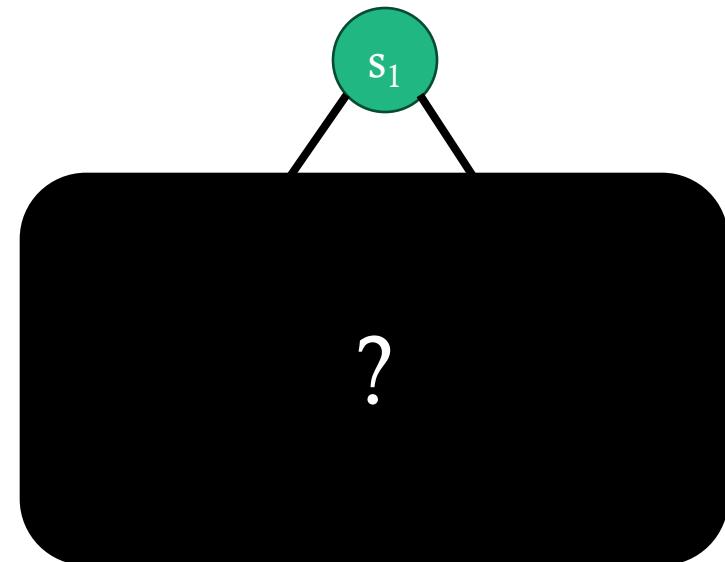


# CATEGORÍAS DE SOLUCIONES

Libres de modelos

Cuando el funcionamiento interno del entorno no es visible para el agente,

*¿cómo aprende a comportarse?*



# CATEGORÍAS DE SOLUCIONES

## Libres de modelos

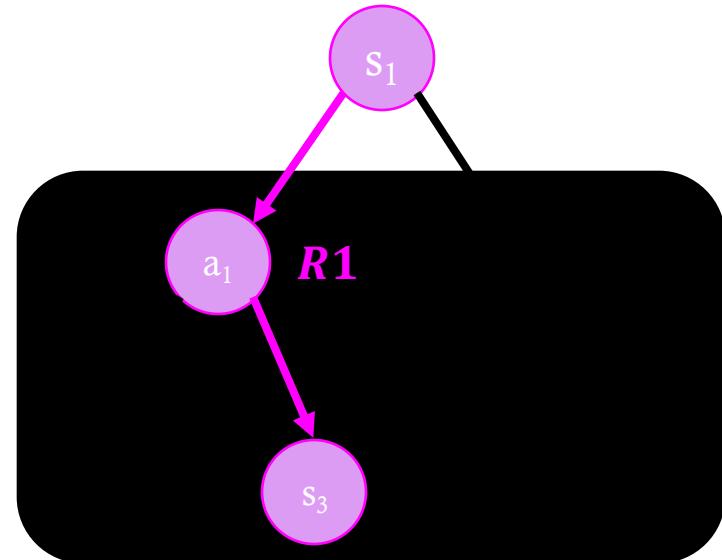
Cuando el funcionamiento interno del entorno no es visible para el agente,

*¿cómo aprende a comportarse?*

👉 A través de la interacción directa con el entorno.

Este enfoque se basa en el famoso método de **prueba y error**: El agente prueba acciones y recibe retroalimentación positiva o negativa.

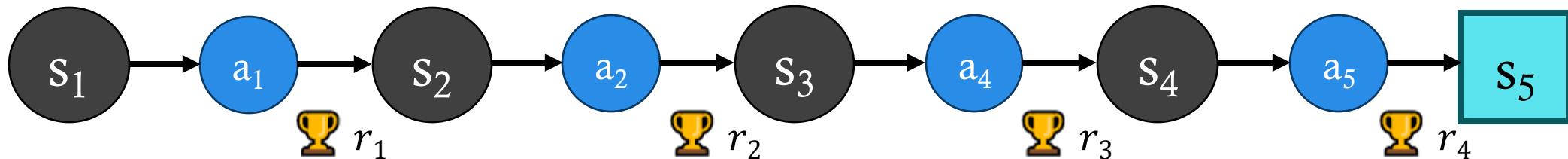
*Este mecanismo es muy similar al modo en que aprenden los seres vivos.*

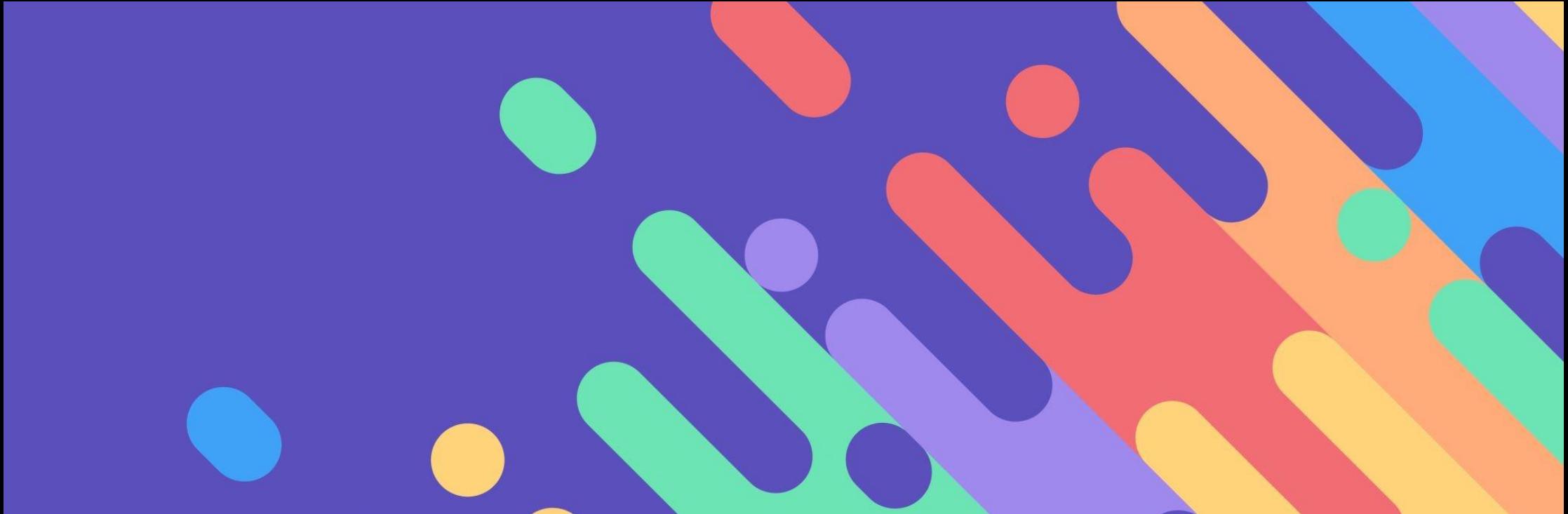


# CATEGORÍAS DE SOLUCIONES

## Libres de modelos

A medida que el agente actúa, sigue una trayectoria. Esa trayectoria, conformada por estados, acciones y recompensas, se convierte en *datos de entrenamiento* para el algoritmo.





---

# ECUACIÓN DE BELLMAN

---

# ECUACIÓN DE BELLMAN

La ecuación de Bellman es un concepto central en el aprendizaje por refuerzo.

Describe cómo el valor de estar en un estado determinado, bajo una política dada, se relaciona con:

- La recompensa inmediata, y
- El valor esperado del siguiente estado.

Para calcular el valor de un estado  $s_t$ ,  $V(s_t)$ , se utiliza la **suma ponderada** de las recompensas futuras (el retorno).

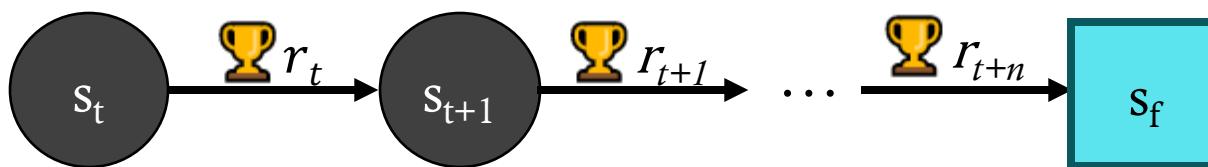
# ECUACIÓN DE BELLMAN

La ecuación de Bellman es un concepto central en el aprendizaje por refuerzo.

Describe cómo el valor de estar en un estado determinado, bajo una política dada, se relaciona con:

- La recompensa inmediata, y
- El valor esperado del siguiente estado.

Para calcular el valor de un estado  $s_t$ ,  $V(s_t)$ , se utiliza la **suma ponderada** de las recompensas futuras (el retorno).



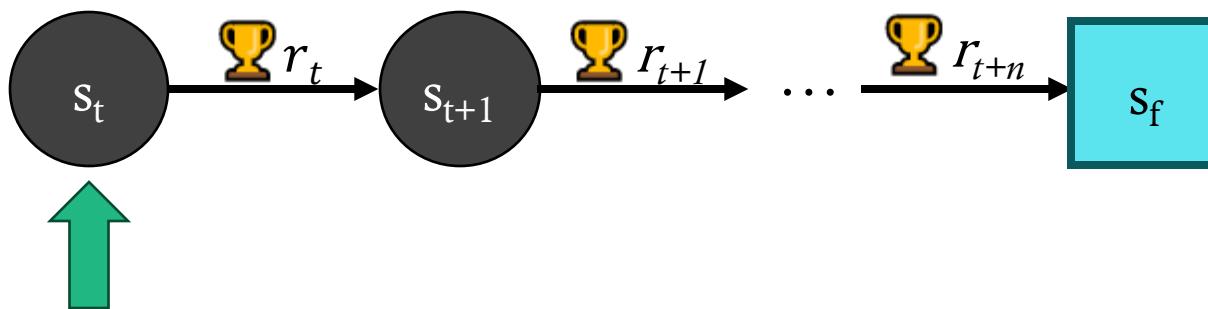
# ECUACIÓN DE BELLMAN

La ecuación de Bellman es un concepto central en el aprendizaje por refuerzo.

Describe cómo el valor de estar en un estado determinado, bajo una política dada, se relaciona con:

- La recompensa inmediata, y
- El valor esperado del siguiente estado.

Para calcular el valor de un estado  $s_t$ ,  $V(s_t)$ , se utiliza la **suma ponderada** de las recompensas futuras (el retorno).



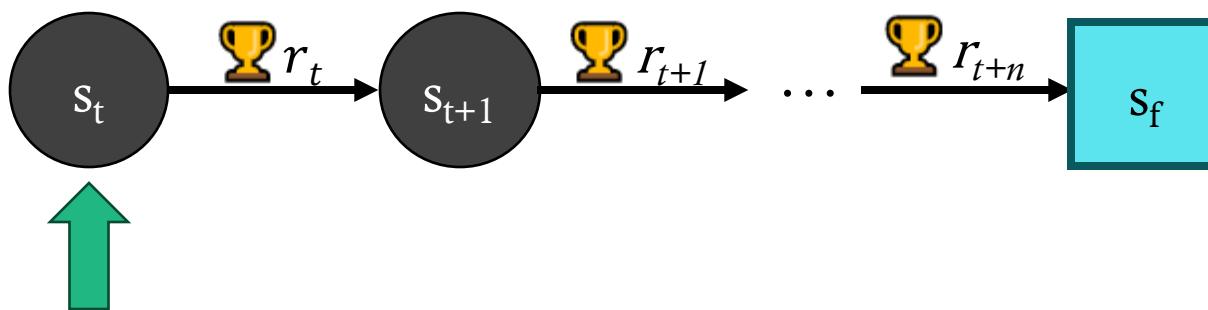
# ECUACIÓN DE BELLMAN

La ecuación de Bellman es un concepto central en el aprendizaje por refuerzo.

Describe cómo el valor de estar en un estado determinado, bajo una política dada, se relaciona con:

- La recompensa inmediata, y
- El valor esperado del siguiente estado.

Para calcular el valor de un estado  $s_t$ ,  $V(s_t)$ , se utiliza la **suma ponderada** de las recompensas futuras (el retorno).



$$Ret = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots + \gamma^{t+n} r_{t+n}$$

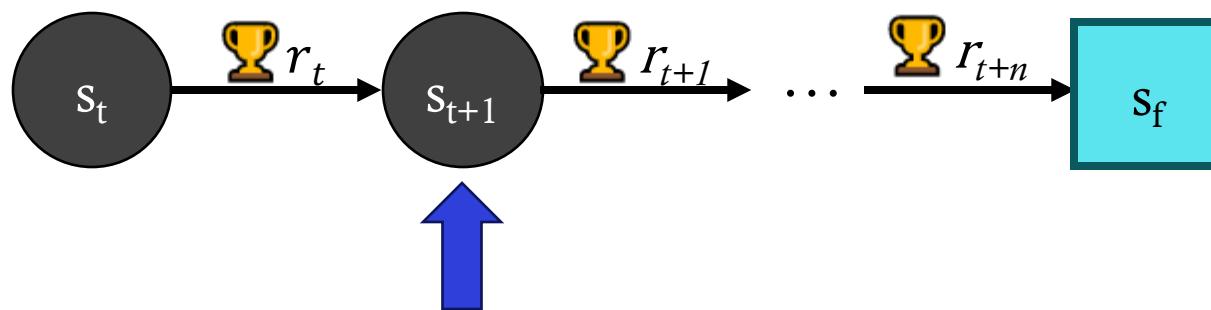
# ECUACIÓN DE BELLMAN

La ecuación de Bellman es un concepto central en el aprendizaje por refuerzo.

Describe cómo el valor de estar en un estado determinado, bajo una política dada, se relaciona con:

- La recompensa inmediata, y
- El valor esperado del siguiente estado.

Para calcular el valor de un estado  $s_t$ ,  $V(s_t)$ , se utiliza la **suma ponderada** de las recompensas futuras (el retorno).



$$Ret_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots + \gamma^{t+n} r_{t+n}$$

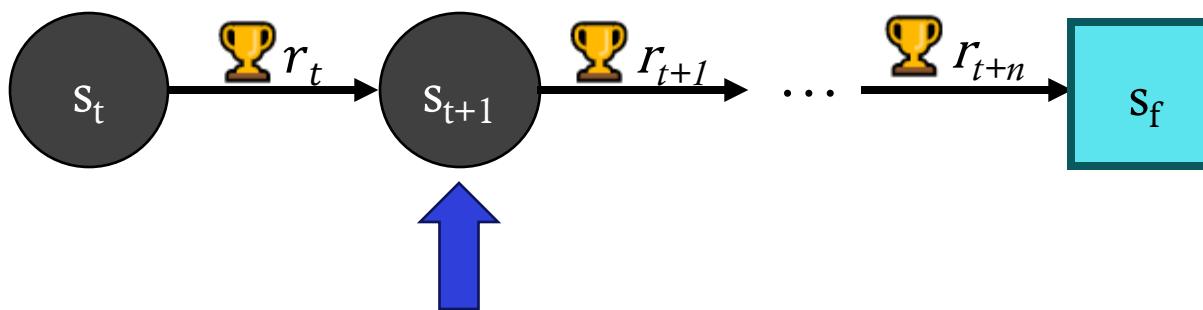
# ECUACIÓN DE BELLMAN

La ecuación de Bellman es un concepto central en el aprendizaje por refuerzo.

Describe cómo el valor de estar en un estado determinado, bajo una política dada, se relaciona con:

- La recompensa inmediata, y
- El valor esperado del siguiente estado.

Para calcular el valor de un estado  $s_t$ ,  $V(s_t)$ , se utiliza la **suma ponderada** de las recompensas futuras (el retorno).



$$Ret_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots + \gamma^{t+n} r_{t+n}$$

$$Ret_{t+1} = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \dots + \gamma^{t+n-1} r_{t+n}$$

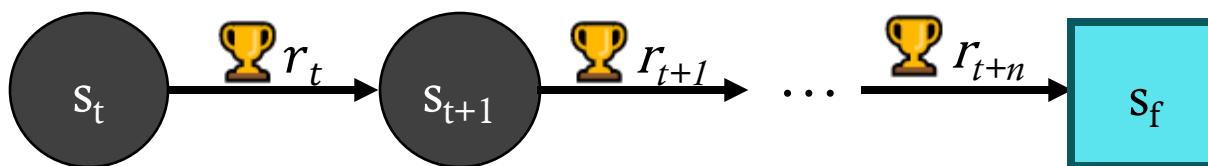
# ECUACIÓN DE BELLMAN

La ecuación de Bellman es un concepto central en el aprendizaje por refuerzo.

Describe cómo el valor de estar en un estado determinado, bajo una política dada, se relaciona con:

- La recompensa inmediata, y
- El valor esperado del siguiente estado.

Para calcular el valor de un estado  $s_t$ ,  $V(s_t)$ , se utiliza la **suma ponderada** de las recompensas futuras (el retorno).



$$Ret_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots + \gamma^{t+n} r_{t+n}$$

$$Ret_{t+1} = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \dots + \gamma^{t+n-1} r_{t+n}$$

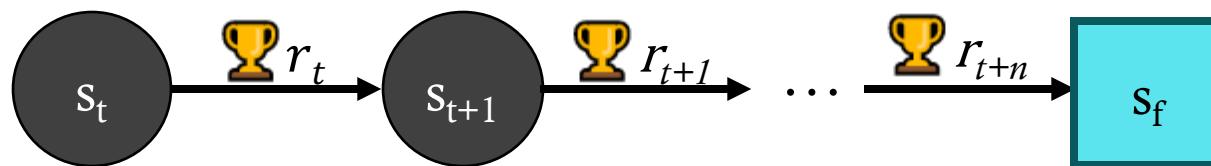
# ECUACIÓN DE BELLMAN

La ecuación de Bellman es un concepto central en el aprendizaje por refuerzo.

Describe cómo el valor de estar en un estado determinado, bajo una política dada, se relaciona con:

- La recompensa inmediata, y
- El valor esperado del siguiente estado.

Para calcular el valor de un estado  $s_t$ ,  $V(s_t)$ , se utiliza la **suma ponderada** de las recompensas futuras (el retorno).



$$Ret_t = r_t + \gamma(r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \dots + \gamma^{t+n-1} r_{t+n})$$

$$Ret_{t+1} = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \dots + \gamma^{t+n-1} r_{t+n}$$

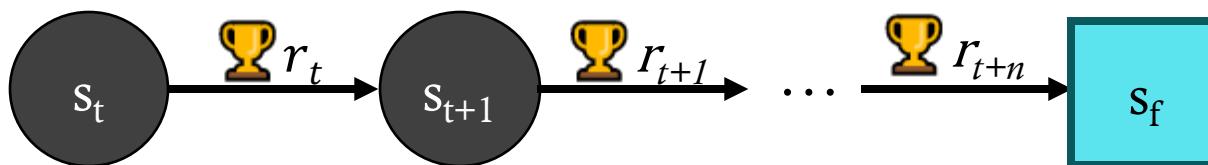
# ECUACIÓN DE BELLMAN

La ecuación de Bellman es un concepto central en el aprendizaje por refuerzo.

Describe cómo el valor de estar en un estado determinado, bajo una política dada, se relaciona con:

- La recompensa inmediata, y
- El valor esperado del siguiente estado.

Para calcular el valor de un estado  $s_t$ ,  $V(s_t)$ , se utiliza la **suma ponderada** de las recompensas futuras (el retorno).



$$Ret_t = r_t + \gamma(r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \dots + \gamma^{t+n-1} r_{t+n})$$

$$Ret_{t+1} = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \dots + \gamma^{t+n-1} r_{t+n}$$

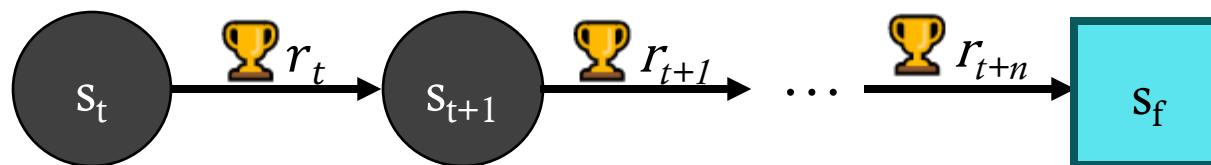
# ECUACIÓN DE BELLMAN

La ecuación de Bellman es un concepto central en el aprendizaje por refuerzo.

Describe cómo el valor de estar en un estado determinado, bajo una política dada, se relaciona con:

- La recompensa inmediata, y
- El valor esperado del siguiente estado.

Para calcular el valor de un estado  $s_t$ ,  $V(s_t)$ , se utiliza la **suma ponderada** de las recompensas futuras (el retorno).



$$Ret_t = r_t + \gamma Ret_{t+1}$$

$$Ret_{t+1} = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \dots + \gamma^{t+n-1} r_{t+n}$$

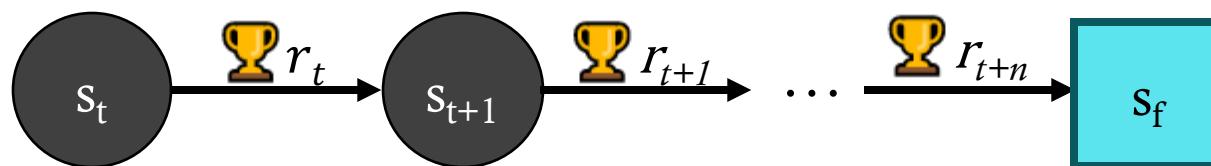
# ECUACIÓN DE BELLMAN

La ecuación de Bellman es un concepto central en el aprendizaje por refuerzo.

Describe cómo el valor de estar en un estado determinado, bajo una política dada, se relaciona con:

- La recompensa inmediata, y
- El valor esperado del siguiente estado.

Para calcular el valor de un estado  $s_t$ ,  $V(s_t)$ , se utiliza la **suma ponderada** de las recompensas futuras (el retorno).



$$Ret_t = r_t + \gamma Ret_{t+1}$$

---

# ECUACIÓN DE BELLMAN

La ecuación tiene una naturaleza recursiva, y eso es algo que podemos aprovechar...

---

# ECUACIÓN DE BELLMAN

La ecuación tiene una naturaleza recursiva, y eso es algo que podemos aprovechar...

usando la **ecuación de Bellman**:

$$V_\pi(s_t) = E_\pi[r_t + \gamma V_\pi(s_{t+1})]$$

Esto nos dice que el valor actual  $V(s_t)$  es la recompensa inmediata  $r_t$  más el valor esperado descontado del siguiente estado  $\gamma V(s_{t+1})$

---

# ECUACIÓN DE BELLMAN

La ecuación tiene una naturaleza recursiva, y eso es algo que podemos aprovechar...

usando la **ecuación de Bellman**:

$$V_{\pi}(s_t) = E_{\pi}[r_t + \gamma V_{\pi}(s_{t+1})]$$

*Bajo la política  $\pi$*

Esto nos dice que el valor actual  $V(s_t)$  es la recompensa inmediata  $r_t$  más el valor esperado descontado del siguiente estado  $\gamma V(s_{t+1})$

---

# ECUACIÓN DE BELLMAN

La ecuación tiene una naturaleza recursiva, y eso es algo que podemos aprovechar...

usando la **ecuación de Bellman**:

$$V_\pi(s_t) = E_\pi[r_t + \gamma V_\pi(s_{t+1})]$$

Esto nos dice que el valor actual  $V(s_t)$  es la recompensa inmediata  $r_t$  más el valor esperado descontado del siguiente estado  $\gamma V(s_{t+1})$

---

# ECUACIÓN DE BELLMAN

La ecuación tiene una naturaleza recursiva, y eso es algo que podemos aprovechar...

usando la **ecuación de Bellman**:

$$V_{\pi}(s_t) = \underline{E_{\pi}}[r_t + \gamma V_{\pi}(s_{t+1})]$$

 *Esperanza matemática*

Esto nos dice que el valor actual  $V(s_t)$  es la recompensa inmediata  $r_t$  más el valor esperado descontado del siguiente estado  $\gamma V(s_{t+1})$

---

# ECUACIÓN DE BELLMAN

La ecuación tiene una naturaleza recursiva, y eso es algo que podemos aprovechar...

usando la **ecuación de Bellman**:

$$V_\pi(s_t) = E_\pi[r_t + \gamma V_\pi(s_{t+1})]$$

Esto nos dice que el valor actual  $V(s_t)$  es la recompensa inmediata  $r_t$  más el valor esperado descontado del siguiente estado  $\gamma V(s_{t+1})$

---

---

# ECUACIÓN DE BELLMAN

La ecuación tiene una naturaleza recursiva, y eso es algo que podemos aprovechar...

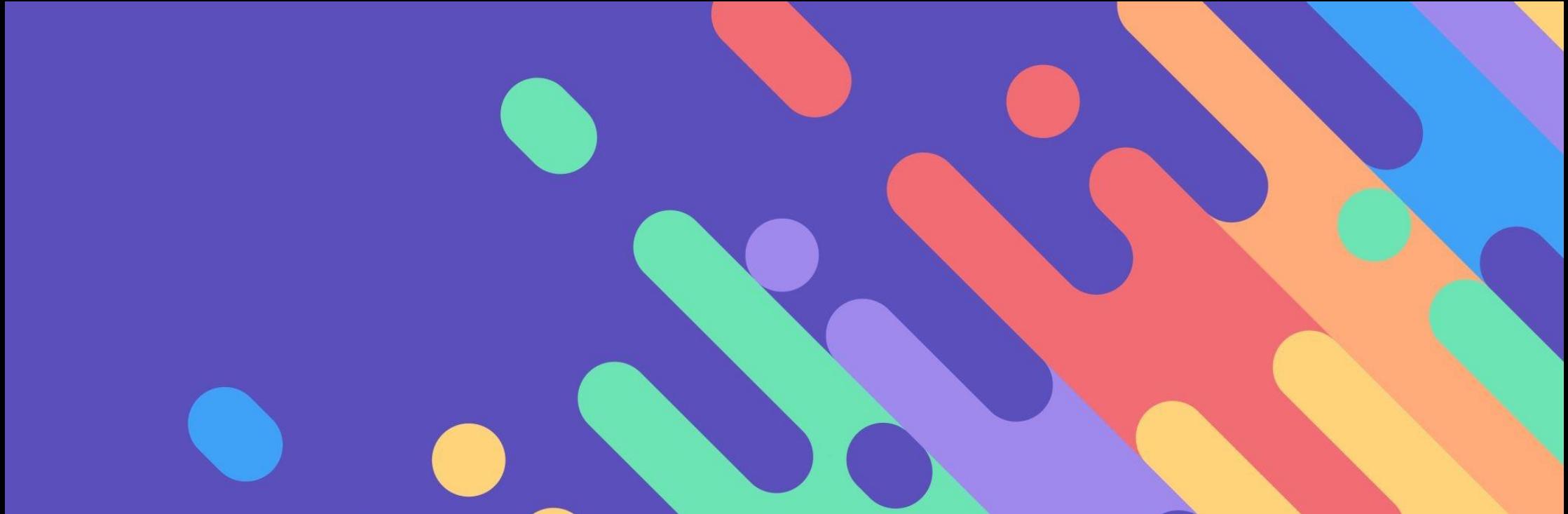
usando la **ecuación de Bellman**:

$$V_\pi(s_t) = E_\pi[r_t + \gamma V_\pi(s_{t+1})]$$

Esto nos dice que el valor actual  $V(s_t)$  es la recompensa inmediata  $r_t$  más el valor esperado descontado del siguiente estado  $\gamma V(s_{t+1})$

👉 *En lugar de calcular manualmente todas las recompensas futuras, estimamos el valor como: Recompensa inmediata + valor del siguiente estado (con descuento)*

---



---

# ESTRATEGIAS DE APRENDIZAJE

---

# ESTRATEGIAS DE APRENDIZAJE

El agente aprende interactuando con el entorno. A partir de la experiencia y las recompensas recibidas, **actualiza su función de valor o su política**.

Las dos estrategias más comunes son:

- **Aprendizaje por diferencia temporal (TD)**: aprende con cada transición de estado.
- **Monte Carlo**: aprende usando la experiencia completa de un episodio antes de actualizar.

---

# ESTRATEGIAS DE APRENDIZAJE

## Aprendizaje por diferencia temporal (TD)

En TD, el agente espera una transición al siguiente estado para actualizar el valor  $V(s_t)$ , usando la recompensa  $r_t$  y el valor del siguiente estado  $V(s_{t+1})$ , como indica la *ecuación de Bellman*...

---

# ESTRATEGIAS DE APRENDIZAJE

## Aprendizaje por diferencia temporal (TD)

En TD, el agente espera una transición al siguiente estado para actualizar el valor  $V(s_t)$ , usando la recompensa  $r_t$  y el valor del siguiente estado  $V(s_{t+1})$ , como indica la *ecuación de Bellman*...

$$V_{new}(s_t) \leftarrow V_{old}(s_t) + \alpha[r_t + \gamma V(s_{t+1}) - V_{old}(s_t)]$$

---

# ESTRATEGIAS DE APRENDIZAJE

## Aprendizaje por diferencia temporal (TD)

En TD, el agente espera una transición al siguiente estado para actualizar el valor  $V(s_t)$ , usando la recompensa  $r_t$  y el valor del siguiente estado  $V(s_{t+1})$ , como indica la *ecuación de Bellman*...

$$V_{new}(s_t) \leftarrow V_{old}(s_t) + \alpha[r_t + \gamma V(s_{t+1}) - V_{old}(s_t)]$$

Ecuación de Bellman:  $V_\pi(s_t) = E_\pi[r_t + \gamma V_\pi(s_{t+1})]$

# ESTRATEGIAS DE APRENDIZAJE

## Aprendizaje por diferencia temporal (TD)

En TD, el agente espera una transición al siguiente estado para actualizar el valor  $V(s_t)$ , usando la recompensa  $r_t$  y el valor del siguiente estado  $V(s_{t+1})$ , como indica la *ecuación de Bellman*...

$$V_{new}(s_t) \leftarrow V_{old}(s_t) + \alpha [r_t + \gamma V(s_{t+1}) - V_{old}(s_t)]$$

Ecuación de Bellman:  $V_\pi(s_t) = E_\pi[r_t + \gamma V_\pi(s_{t+1})]$

# ESTRATEGIAS DE APRENDIZAJE

## Aprendizaje por diferencia temporal (TD)

En TD, el agente espera una transición al siguiente estado para actualizar el valor  $V(s_t)$ , usando la recompensa  $r_t$  y el valor del siguiente estado  $V(s_{t+1})$ , como indica la *ecuación de Bellman*...

$$V_{new}(s_t) \leftarrow V_{old}(s_t) + \alpha [r_t + \gamma V(s_{t+1}) - V_{old}(s_t)]$$

Ecuación de Bellman:  $V_\pi(s_t) = E_\pi[r_t + \gamma V_\pi(s_{t+1})]$

---

# ESTRATEGIAS DE APRENDIZAJE

## Aprendizaje por diferencia temporal (TD)

En TD, el agente espera una transición al siguiente estado para actualizar el valor  $V(s_t)$ , usando la recompensa  $r_t$  y el valor del siguiente estado  $V(s_{t+1})$ , como indica la *ecuación de Bellman*...

$$V_{new}(s_t) \leftarrow V_{old}(s_t) + \alpha[r_t + \gamma V(s_{t+1}) - V_{old}(s_t)]$$

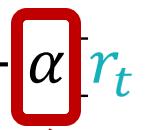
Ecuación de Bellman:  $V_\pi(s_t) = E_\pi[r_t + \gamma V_\pi(s_{t+1})]$

# ESTRATEGIAS DE APRENDIZAJE

## Aprendizaje por diferencia temporal (TD)

En TD, el agente espera una transición al siguiente estado para actualizar el valor  $V(s_t)$ , usando la recompensa  $r_t$  y el valor del siguiente estado  $V(s_{t+1})$ , como indica la *ecuación de Bellman*...

$$V_{new}(s_t) \leftarrow V_{old}(s_t) + \alpha [r_t + \gamma V(s_{t+1}) - V_{old}(s_t)]$$

 Constante de aprendizaje

$$\text{Ecuación de Bellman: } V_\pi(s_t) = E_\pi[r_t + \gamma V_\pi(s_{t+1})]$$

---

# ESTRATEGIAS DE APRENDIZAJE

## Aprendizaje por diferencia temporal (TD)

En TD, el agente espera una transición al siguiente estado para actualizar el valor  $V(s_t)$ , usando la recompensa  $r_t$  y el valor del siguiente estado  $V(s_{t+1})$ , como indica la *ecuación de Bellman*...

$$V_{new}(s_t) \leftarrow V_{old}(s_t) + \alpha[r_t + \gamma V(s_{t+1}) - V_{old}(s_t)]$$

Ecuación de Bellman:  $V_\pi(s_t) = E_\pi[r_t + \gamma V_\pi(s_{t+1})]$

---

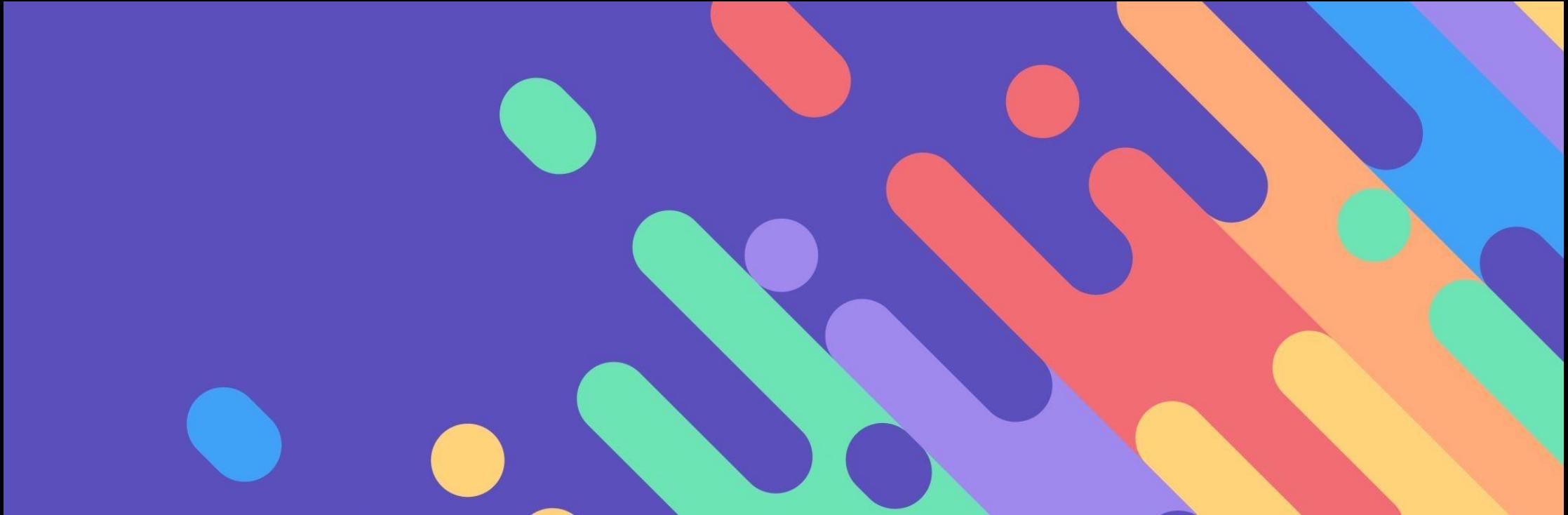
# ESTRATEGIAS DE APRENDIZAJE

## Monte Carlo

En esta técnica, el agente espera hasta que finalice el episodio, calcula el retorno total  $\mathbf{G}_t$ , y lo utiliza para ajustar el valor del estado  $V(s_t)$ :

$$V_{new}(s_t) \leftarrow V_{old}(s_t) + \alpha[\mathbf{G}_t - V_{old}(s_t)]$$

*A diferencia del TD, Monte Carlo no usa la ecuación de Bellman:  
Estima los valores promedio de los estados a partir de episodios completos.*



---

# Q-LEARNING

# Q-LEARNING

Existen muchos algoritmos basados en valores y en políticas, pero todos se pueden reducir a unos pocos principios fundamentales.

A un nivel general, todos comienzan con una estimación inicial y la ajustan con el tiempo a través de iteraciones.

Este proceso incluye cuatro pasos básicos:

1. **Iniciar estimaciones**
2. **Tomar una o más acciones**
3. **Recibir retroalimentación del entorno**
4. **Mejorar las estimaciones**



---

# Q-LEARNING

Q-Learning es un algoritmo basado en valores que se caracteriza por:

- Aprender la función de **valor estado-acción (Q-valor)**.
- Encontrar la política óptima de forma indirecta, mediante el entrenamiento de esta función.
- Ser un enfoque de **aprendizaje por diferencia temporal (TD)**.

# Q-LEARNING

Internamente, la función **Q** se representa mediante una tabla **Q**, donde cada celda corresponde a un par estado-acción:

Q	a <sub>1</sub>	a <sub>2</sub>
s <sub>1</sub>	12	0.3
s <sub>2</sub>	43	12
s <sub>3</sub>	10	93
s <sub>4</sub>	0	5

# Q-LEARNING

---

El algoritmo comienza **inicializando la tabla Q**, generalmente asignando cero a todos los pares estado-acción:

Q	a <sub>1</sub>	a <sub>2</sub>
s <sub>1</sub>	0	0
s <sub>2</sub>	0	0
s <sub>3</sub>	0	0
s <sub>4</sub>	0	0

---

# Q-LEARNING

A medida que el agente interactúa con el entorno, recibe una recompensa y actualiza **la tabla Q** utilizando una versión del aprendizaje por diferencia temporal:

$$Q(s_i, a_j) = Q(s_i, a_j) + \alpha [(r_{ij} + \gamma \max(Q(s_k, :))) - Q(s_i, a_j)]$$

# Q-LEARNING

Veamos como funciona con un ejemplo:

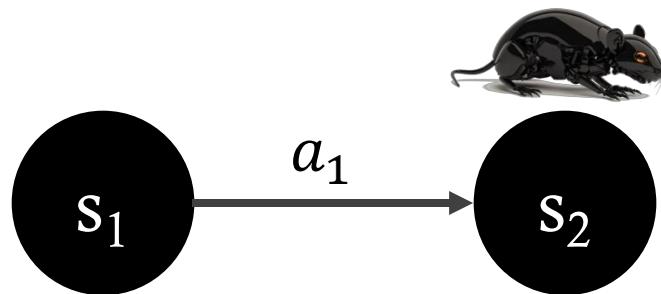


Q	a <sub>1</sub>	a <sub>2</sub>
$s_1$	1.2	0
$s_2$	1	0.3
$s_3$	0	0
$s_4$	0	0

$$\gamma = 0.9$$
$$\alpha = 0.1$$

# Q-LEARNING

Veamos como funciona con un ejemplo:

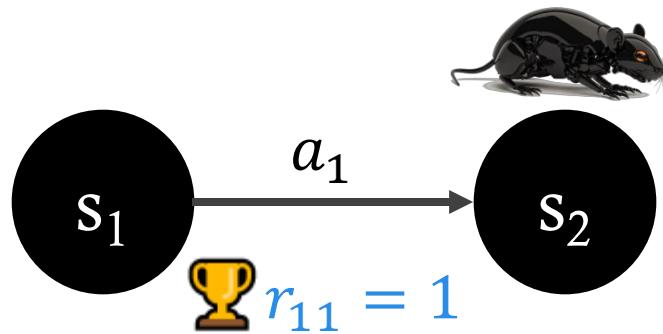


Q	a <sub>1</sub>	a <sub>2</sub>
s <sub>1</sub>	1.2	0
s <sub>2</sub>	1	0.3
s <sub>3</sub>	0	0
s <sub>4</sub>	0	0

$$\gamma = 0.9$$
$$\alpha = 0.1$$

# Q-LEARNING

Veamos como funciona con un ejemplo:

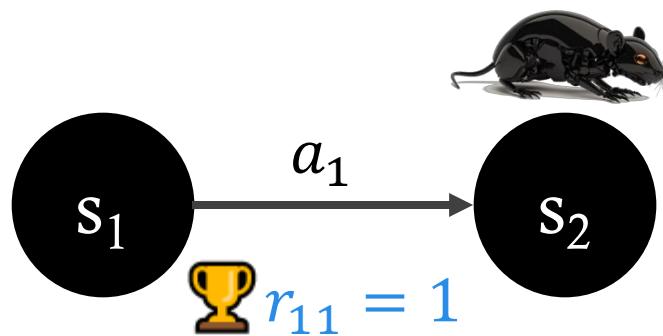


Q	a <sub>1</sub>	a <sub>2</sub>
$s_1$	1.2	0
$s_2$	1	0.3
$s_3$	0	0
$s_4$	0	0

$$\gamma = 0.9$$
$$\alpha = 0.1$$

# Q-LEARNING

Veamos como funciona con un ejemplo:



$$Q(s_1, a_1) = Q(s_1, a_1) + \alpha [(r_{11} + \gamma \max(Q(s_2, :))) - Q(s_1, a_1)]$$

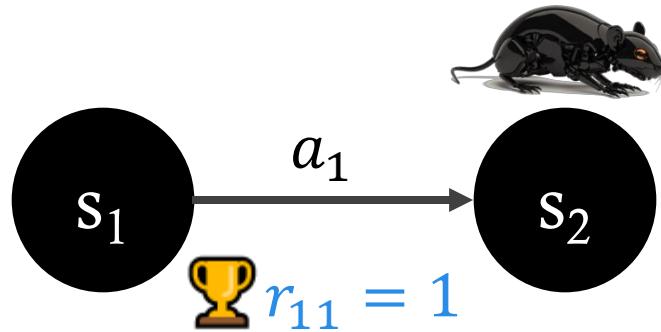


Q	a <sub>1</sub>	a <sub>2</sub>
s <sub>1</sub>	1.2	0
s <sub>2</sub>	1	0.3
s <sub>3</sub>	0	0
s <sub>4</sub>	0	0

$$\gamma = 0.9$$
$$\alpha = 0.1$$

# Q-LEARNING

Veamos como funciona con un ejemplo:



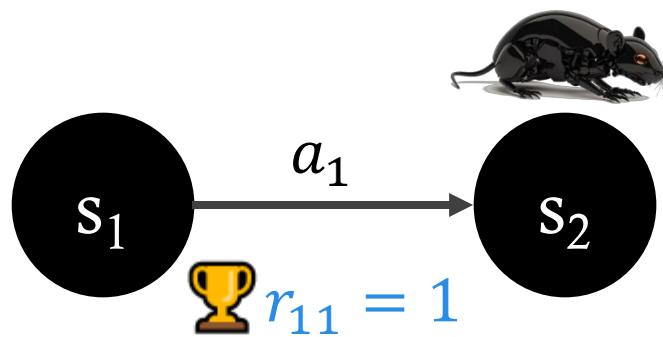
Q	a <sub>1</sub>	a <sub>2</sub>
$s_1$	1.2	0
$s_2$	1	0.3
$s_3$	0	0
$s_4$	0	0

$$\gamma = 0.9$$
$$\alpha = 0.1$$

$$Q(s_1, a_1) = \boxed{Q(s_1, a_1)} + \alpha [(r_{11} + \gamma \max(Q(s_2, :))) - \boxed{Q(s_1, a_1)}]$$

# Q-LEARNING

Veamos como funciona con un ejemplo:



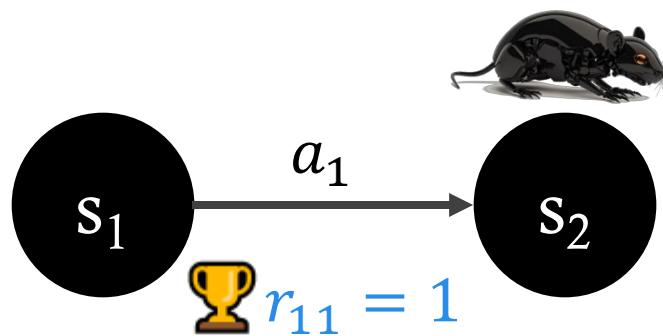
$$Q(s_1, a_1) = \boxed{1.2} + \alpha [(r_{11} + \gamma \max(Q(s_2, :))) - \boxed{1.2}]$$

Q	a <sub>1</sub>	a <sub>2</sub>
$s_1$	1.2	0
$s_2$	1	0.3
$s_3$	0	0
$s_4$	0	0

$$\begin{aligned}\gamma &= 0.9 \\ \alpha &= 0.1\end{aligned}$$

# Q-LEARNING

Veamos como funciona con un ejemplo:



$$Q(s_1, a_1) = \textcolor{teal}{1.2} + \alpha [(\textcolor{blue}{r}_{11} + \gamma \max(Q(s_2, :))) - \textcolor{teal}{1.2}]$$

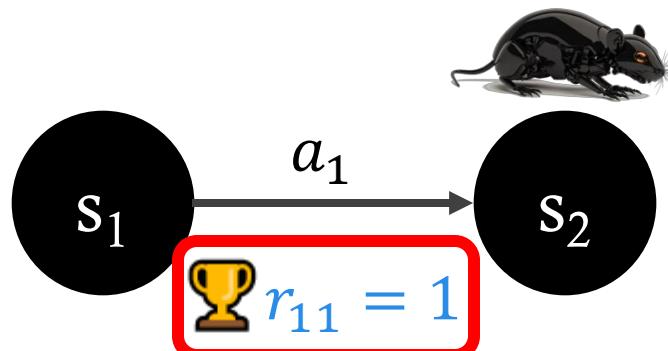


Q	a <sub>1</sub>	a <sub>2</sub>
$s_1$	1.2	0
$s_2$	1	0.3
$s_3$	0	0
$s_4$	0	0

$$\begin{aligned}\gamma &= 0.9 \\ \alpha &= 0.1\end{aligned}$$

# Q-LEARNING

Veamos como funciona con un ejemplo:



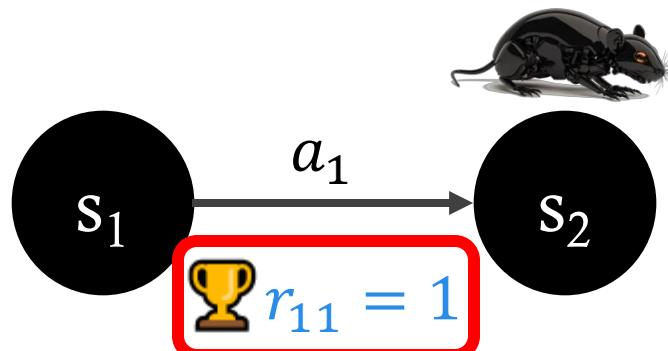
Q	a <sub>1</sub>	a <sub>2</sub>
$s_1$	1.2	0
$s_2$	1	0.3
$s_3$	0	0
$s_4$	0	0

$$\gamma = 0.9$$
$$\alpha = 0.1$$

$$Q(s_1, a_1) = \quad 1.2 \quad + \alpha [(r_{11} + \gamma \max(Q(s_2, :))) - \quad 1.2 \quad ]$$

# Q-LEARNING

Veamos como funciona con un ejemplo:



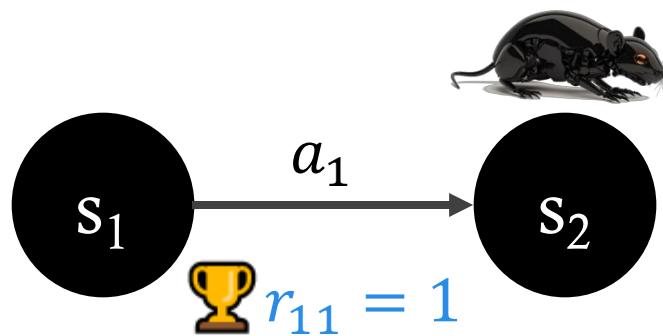
Q	$a_1$	$a_2$
$s_1$	1.2	0
$s_2$	1	0.3
$s_3$	0	0
$s_4$	0	0

$$\gamma = 0.9$$
$$\alpha = 0.1$$

$$Q(s_1, a_1) = \quad 1.2 \quad + \alpha [ \quad 1 \quad + \gamma \max(Q(s_2, :)) \quad - \quad 1.2 \quad ]$$

# Q-LEARNING

Veamos como funciona con un ejemplo:



$$Q(s_1, a_1) = \textcolor{teal}{1.2} + \alpha [(\textcolor{blue}{1} + \gamma \max(Q(s_2, :))) - \textcolor{teal}{1.2}]$$

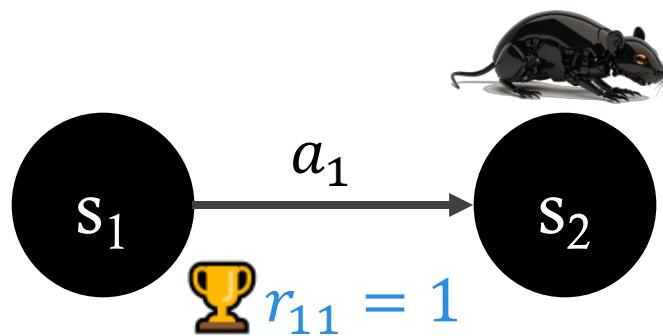


Q	a <sub>1</sub>	a <sub>2</sub>
$s_1$	1.2	0
$s_2$	1	0.3
$s_3$	0	0
$s_4$	0	0

$$\gamma = 0.9$$
$$\alpha = 0.1$$

# Q-LEARNING

Veamos como funciona con un ejemplo:



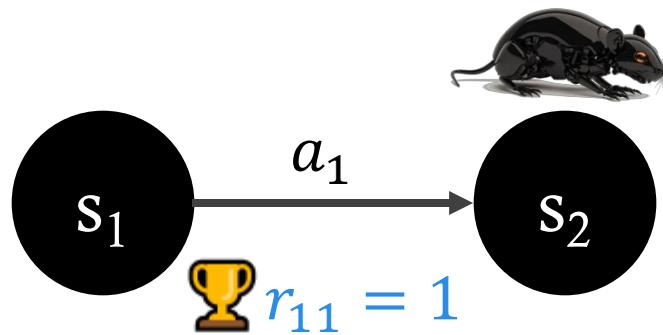
$$Q(s_1, a_1) = \quad 1.2 \quad + \alpha [ (1 - \gamma \max(Q(s_2, :))) - \quad 1.2 \quad ]$$

Q	a <sub>1</sub>	a <sub>2</sub>
$s_1$	1.2	0
$s_2$	1	0.3
$s_3$	0	0
$s_4$	0	0

$$\begin{aligned}\gamma &= 0.9 \\ \alpha &= 0.1\end{aligned}$$

# Q-LEARNING

Veamos como funciona con un ejemplo:



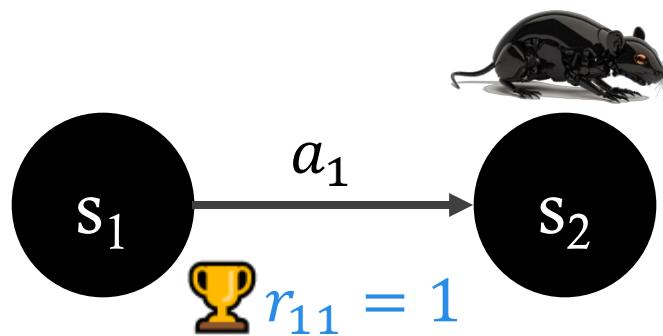
$$Q(s_1, a_1) = \quad 1.2 + 0.1 ( 1 + 0.9 \max(Q(s_2, :)) ) - \quad 1.2 \quad ]$$

Q	a <sub>1</sub>	a <sub>2</sub>
$s_1$	1.2	0
$s_2$	1	0.3
$s_3$	0	0
$s_4$	0	0

$$\begin{aligned}\gamma &= 0.9 \\ \alpha &= 0.1\end{aligned}$$

# Q-LEARNING

Veamos como funciona con un ejemplo:



$$Q(s_1, a_1) = \textcolor{teal}{1.2} + 0.1 [(\textcolor{blue}{1} + 0.9 \max(Q(s_2, :))) - \textcolor{teal}{1.2}]$$

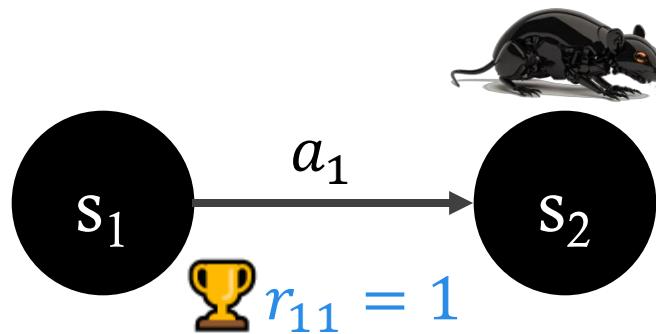


Q	a <sub>1</sub>	a <sub>2</sub>
$s_1$	1.2	0
$s_2$	1	0.3
$s_3$	0	0
$s_4$	0	0

$$\gamma = 0.9$$
$$\alpha = 0.1$$

# Q-LEARNING

Veamos como funciona con un ejemplo:



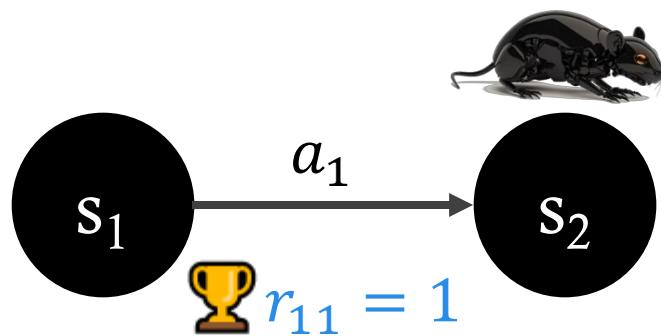
Q	a <sub>1</sub>	a <sub>2</sub>
s <sub>1</sub>	1.2	0
s <sub>2</sub>	1	0.3
s <sub>3</sub>	0	0
s <sub>4</sub>	0	0

$$\gamma = 0.9$$
$$\alpha = 0.1$$

$$Q(s_1, a_1) = \quad 1.2 + 0.1 [( \ 1 + 0.9 \max(Q(s_2, :)) ) - \quad 1.2 \quad ]$$

# Q-LEARNING

Veamos como funciona con un ejemplo:



$$Q(s_1, a_1) = \textcolor{teal}{1.2} + 0.1 [(\textcolor{blue}{1} + 0.9 \max(Q(s_2, :))) - \textcolor{teal}{1.2}]$$

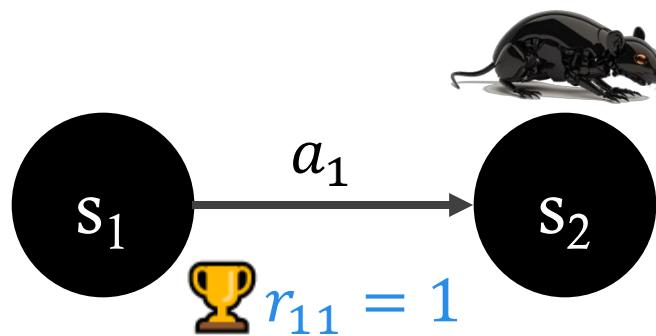


Q	a <sub>1</sub>	a <sub>2</sub>
s <sub>1</sub>	1.2	0
s <sub>2</sub>	1	0.3
s <sub>3</sub>	0	0
s <sub>4</sub>	0	0

$$\gamma = 0.9$$
$$\alpha = 0.1$$

# Q-LEARNING

Veamos como funciona con un ejemplo:



$$Q(s_1, a_1) = \textcolor{teal}{1.2} + 0.1 [(\textcolor{blue}{1} + 0.9 \textcolor{red}{1}) - \textcolor{teal}{1.2}]$$

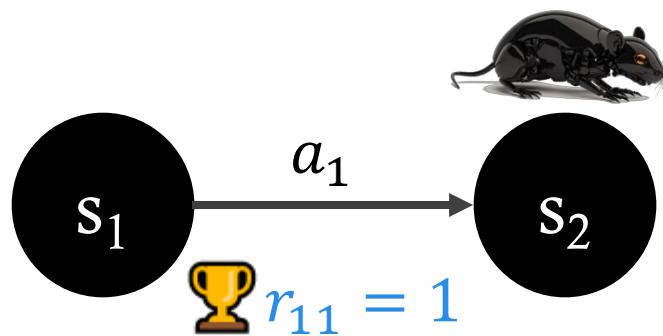


Q	a <sub>1</sub>	a <sub>2</sub>
s <sub>1</sub>	1.2	0
s <sub>2</sub>	1	0.3
s <sub>3</sub>	0	0
s <sub>4</sub>	0	0

$$\gamma = 0.9$$
$$\alpha = 0.1$$

# Q-LEARNING

Veamos como funciona con un ejemplo:



$$Q(s_1, a_1) = \textcolor{teal}{1.2} + 0.1 [(\textcolor{blue}{1} + 0.9 \textcolor{blue}{1}) - \textcolor{teal}{1.2}]$$

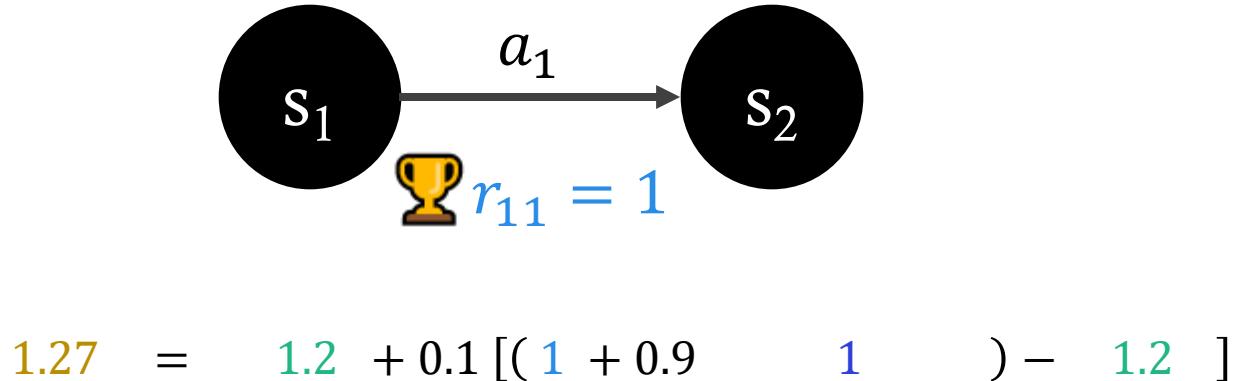


Q	$a_1$	$a_2$
$s_1$	1.2	0
$s_2$	1	0.3
$s_3$	0	0
$s_4$	0	0

$$\gamma = 0.9$$
$$\alpha = 0.1$$

# Q-LEARNING

Veamos como funciona con un ejemplo:

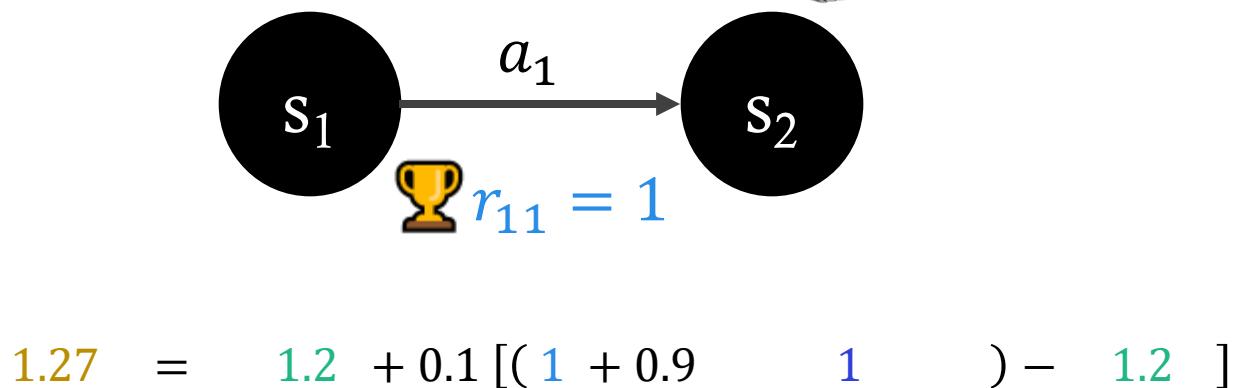


Q	$a_1$	$a_2$
$s_1$	1.2	0
$s_2$	1	0.3
$s_3$	0	0
$s_4$	0	0

$$\gamma = 0.9$$
$$\alpha = 0.1$$

# Q-LEARNING

Veamos como funciona con un ejemplo:

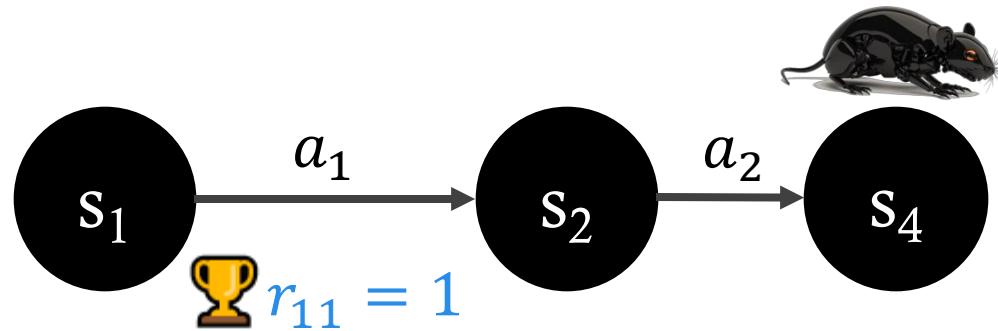


Q	a <sub>1</sub>	a <sub>2</sub>
$s_1$	1.27	0
$s_2$	1	0.3
$s_3$	0	0
$s_4$	0	0

$$\gamma = 0.9$$
$$\alpha = 0.1$$

# Q-LEARNING

Veamos como funciona con un ejemplo:



Q	a <sub>1</sub>	a <sub>2</sub>
s <sub>1</sub>	1.27	0
s <sub>2</sub>	1	0.3
s <sub>3</sub>	0	0
s <sub>4</sub>	0	0

$$\gamma = 0.9$$
$$\alpha = 0.1$$

---

# Q-LEARNING

Tomar una acción

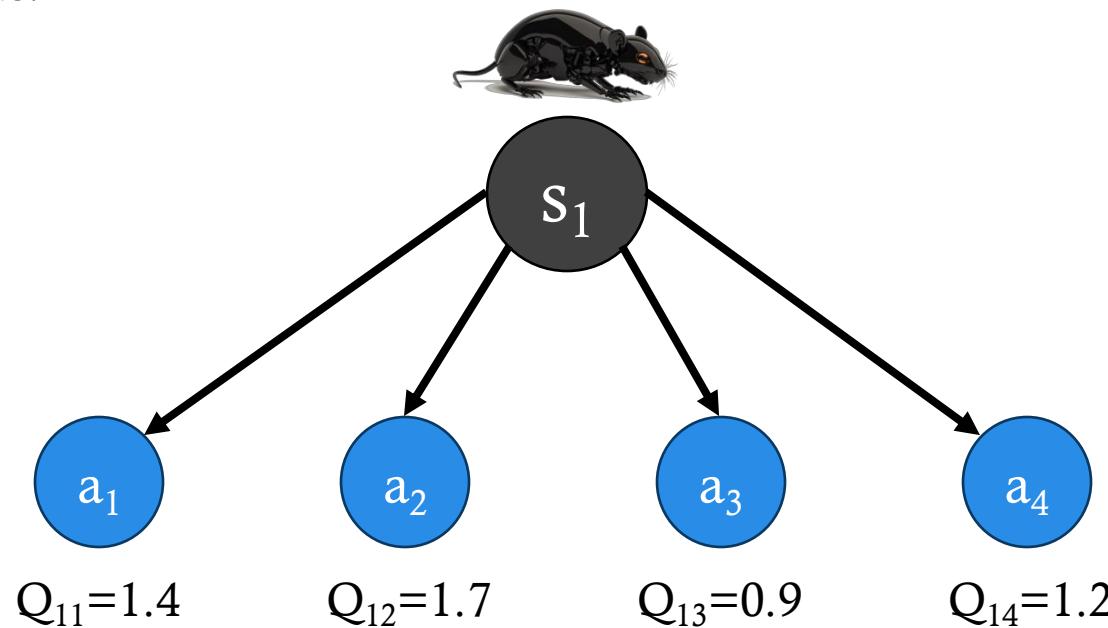
¿Cómo decide el agente qué acción realizar?

El agente debe equilibrar *exploración* (probar acciones nuevas) y *explotación* (usar el conocimiento actual para obtener la mayor recompensa).

# Q-LEARNING

## Exploración

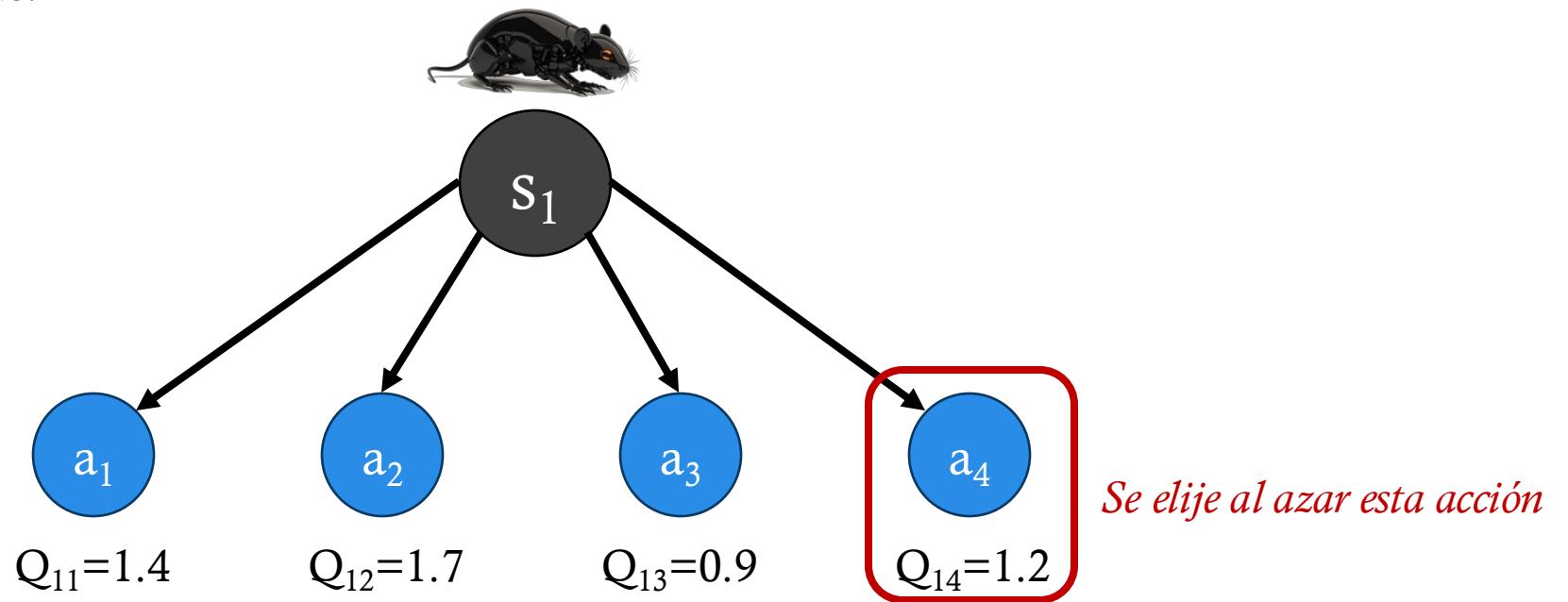
Al inicio del entrenamiento, el agente no sabe qué acciones son *buenas* o *malas*. Durante la *exploración*, prueba diferentes acciones al azar y observa las recompensas, lo que le permite descubrir nuevas estrategias.



# Q-LEARNING

## Exploración

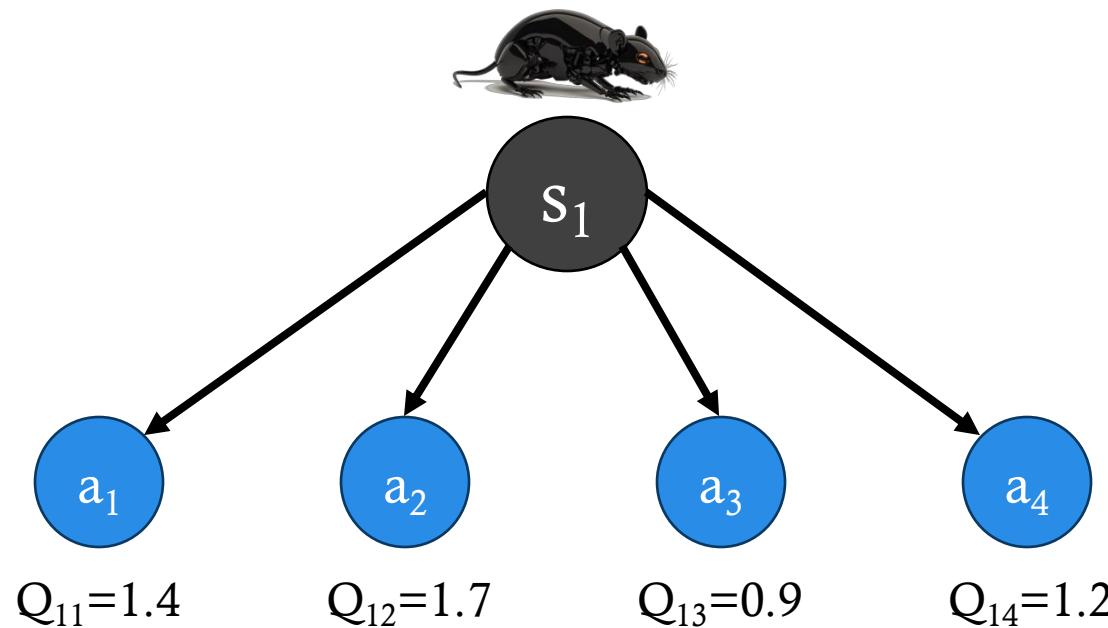
Al inicio del entrenamiento, el agente no sabe qué acciones son *buenas* o *malas*. Durante la *exploración*, prueba diferentes acciones al azar y observa las recompensas, lo que le permite descubrir nuevas estrategias.



# Q-LEARNING

## Explotación

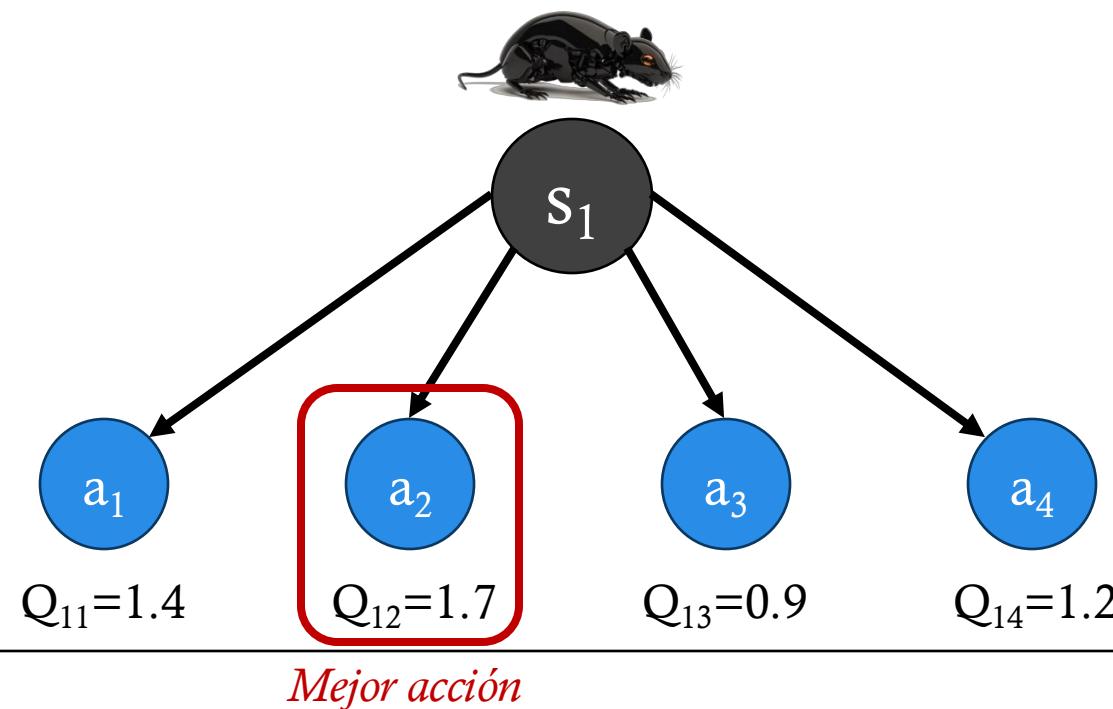
Con el tiempo, el agente acumula conocimiento. Cuando el modelo ya ha aprendido lo suficiente, puede elegir sistemáticamente las mejores acciones, **aquellas que producen el mayor retorno esperado.**



# Q-LEARNING

## Explotación

Con el tiempo, el agente acumula conocimiento. Cuando el modelo ya ha aprendido lo suficiente, puede elegir sistemáticamente las mejores acciones, **aquellas que producen el mayor retorno esperado.**



# Q-LEARNING

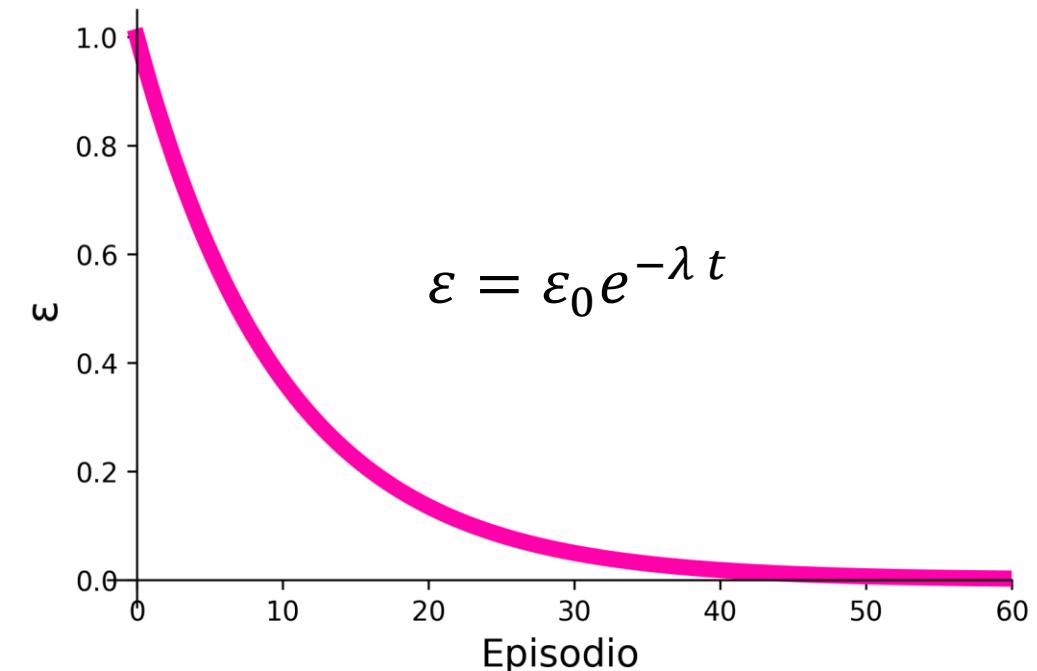
## Exploración vs. Explotación: estrategia $\varepsilon$ -Greedy

Para decidir entre explorar o explotar, se utiliza la estrategia  **$\varepsilon$ -Greedy**.

- Se define una tasa de exploración  $\varepsilon$ , que disminuye gradualmente a lo largo del entrenamiento.
- Al principio, se favorece la *exploración*; luego, se prefiere la *explotación*.



Si  $U[0,1] < \varepsilon \rightarrow$  Explorar  
Si  $U[0,1] > \varepsilon \rightarrow$  Explotar

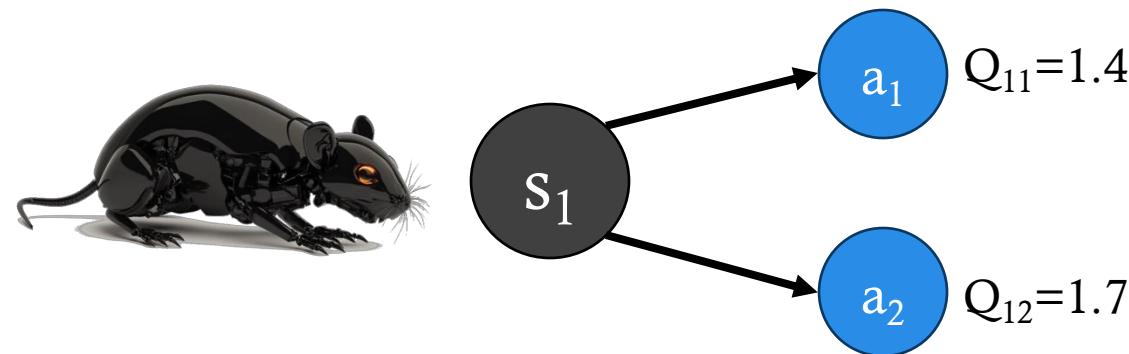


# Q-LEARNING

¿Cómo se elige la política a partir del valor Q?

Al finalizar el entrenamiento, se asume que el agente ha aprendido. La política óptima  $\pi^*$  se determina eligiendo la acción con mayor valor Q en cada estado:

$$\pi^*(s) = \operatorname{argmax}(Q(s, a))$$

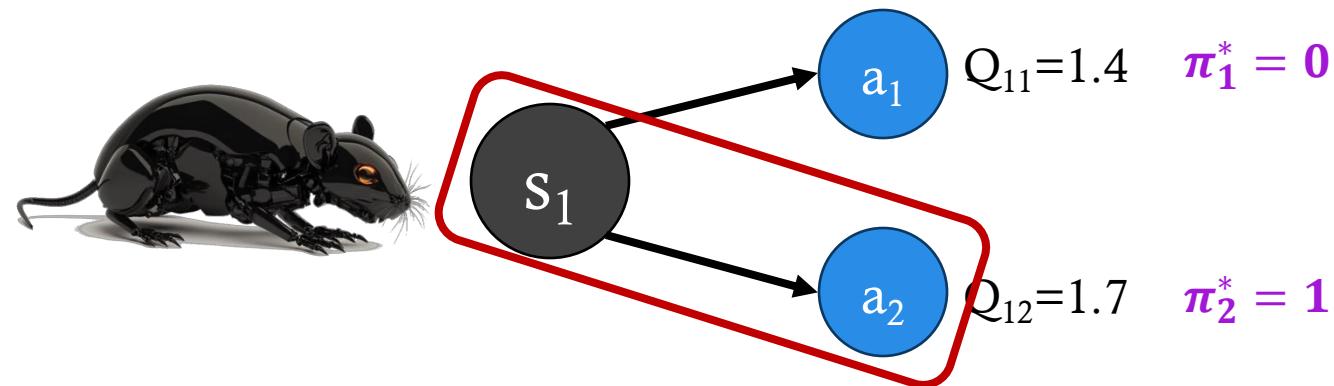


# Q-LEARNING

¿Cómo se elige la política a partir del valor Q?

Al finalizar el entrenamiento, se asume que el agente ha aprendido. La política óptima  $\pi^*$  se determina eligiendo la acción con mayor valor Q en cada estado:

$$\pi^*(s) = \operatorname{argmax}(Q(s, a))$$



---

# Q-LEARNING

¿Cómo se elige la política a partir del valor Q?

Generalmente, la política óptima es **determinista**, eligiendo siempre la mejor acción.

Sin embargo, puede ser estocástica si hay empate entre **dos acciones**. En ese caso, se elige aleatoriamente entre ellas con igual probabilidad.

*En entornos con oponentes, una política estocástica es útil para evitar ser predecible, dificultando que el adversario explote el comportamiento del agente.*

---

# Q-LEARNING

## Otros algoritmos: similitudes y diferencias

Aunque algunos algoritmos se basan en valores y otros directamente en políticas, comparten ciertas características clave:

- **Frecuencia de actualización:**

- Por episodio.
- Por cada paso
- Cada n pasos

- **Profundidad de actualización:**

- Desde el final del episodio
- Paso a paso
- n pasos hacia atrás

- **Fórmula de actualización:**

- Algunas variantes usan versiones distintas de la ecuación de Bellman.
- En algoritmos basados en políticas, se ajustan las probabilidades de acción en función de las recompensas recibidas.