

2

---

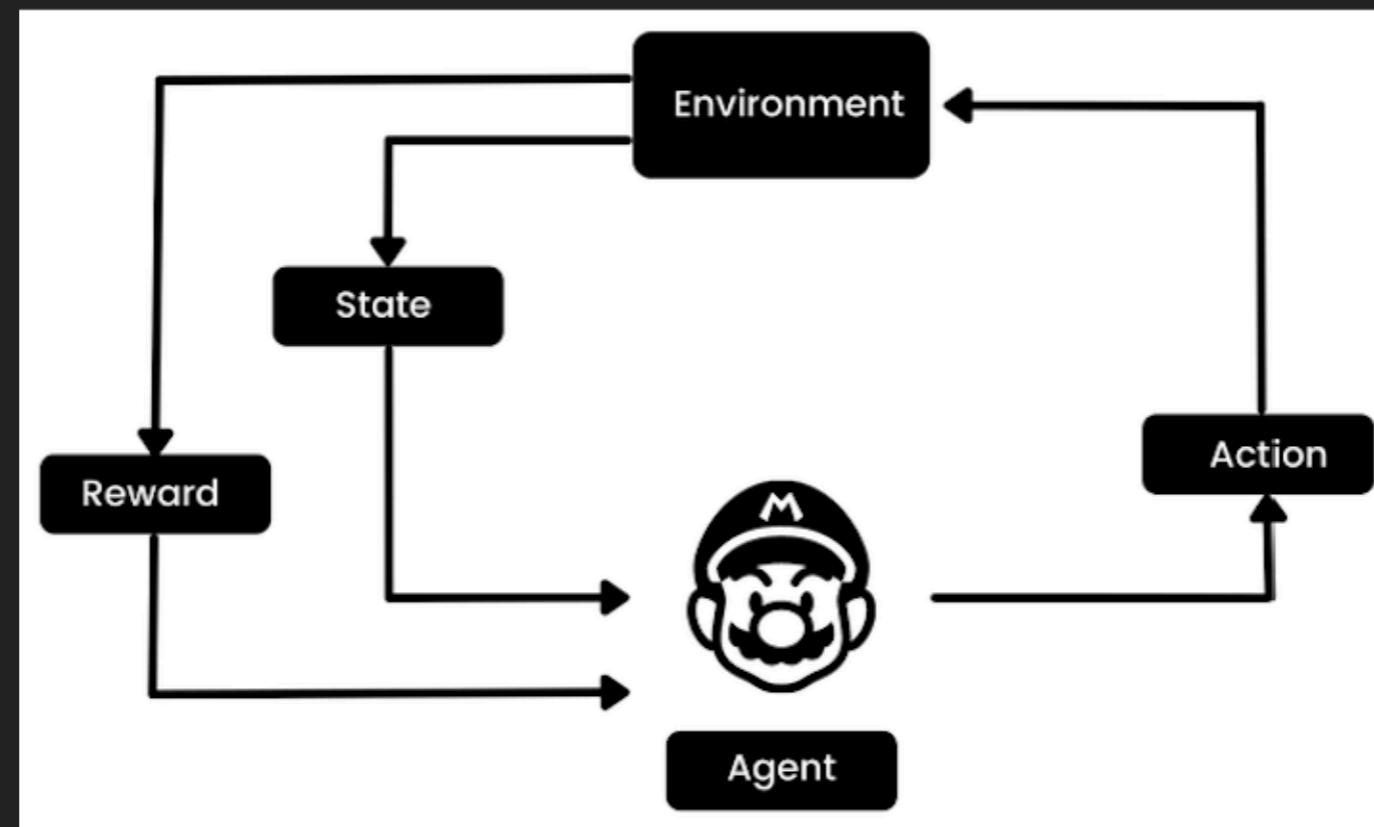
# FUNDAMENTOS DEL APRENDIZAJE POR REFUERZO

# ¿QUÉ ES APRENDIZAJE POR REFUERZO?



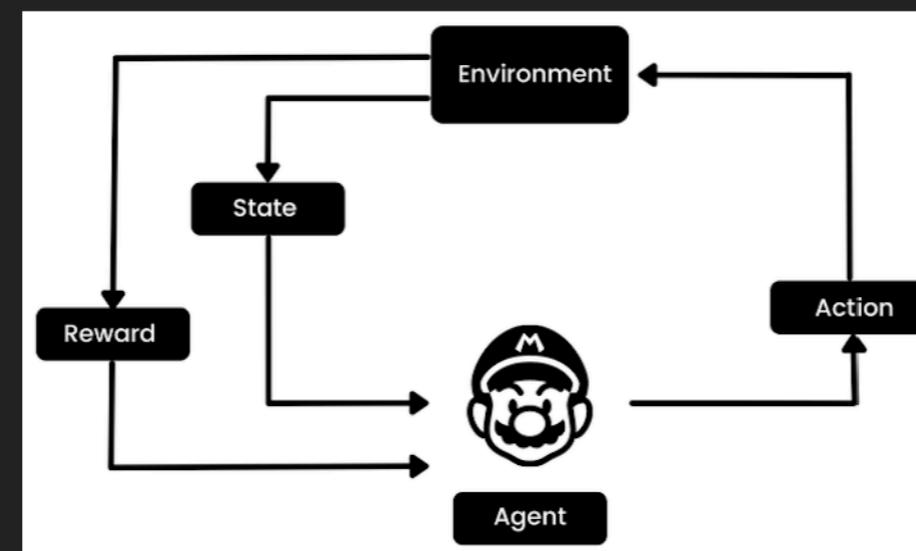
# ¿QUÉ ES APRENDIZAJE POR REFUERZO?

- ▶ Aprendizaje por refuerzo (o Reinforcement learning) es una técnica de Machine Learning que consiste en un algoritmo en el que un **agente** debe aprender un comportamiento a través de interacciones de prueba y error con un **entorno**.



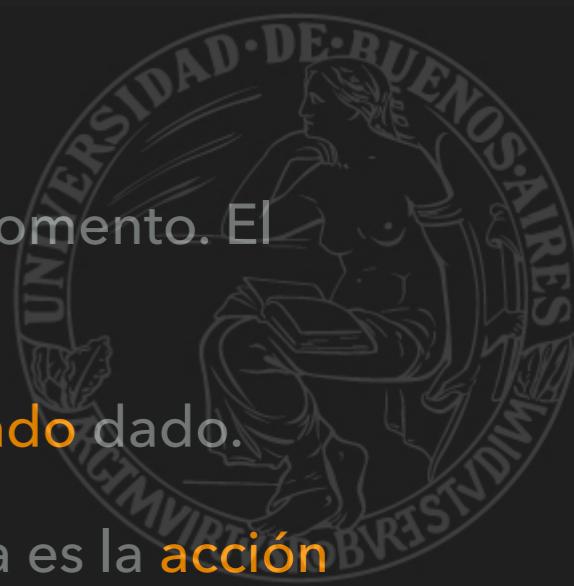
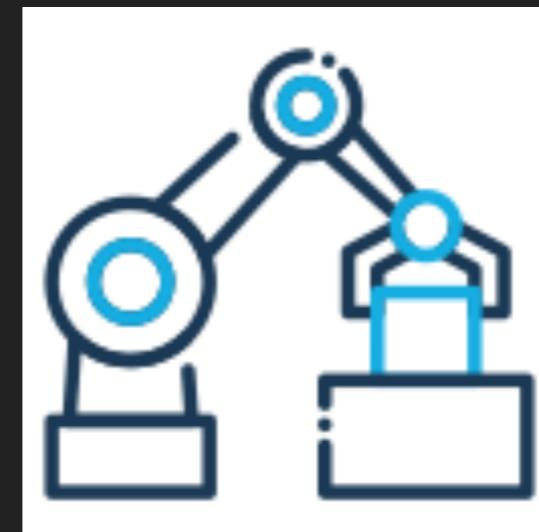
# ¿QUÉ ES APRENDIZAJE POR REFUERZO?

- ▶ Por cada **acción** que realiza el agente recibe:
  - ✓ Puntaje positivo (o **recompensa**) si la acción es correcta.
  - ✓ Puntaje negativo (o **penalización**) si comete un **error**.
- ▶ El objetivo del agente es **maximizar la suma de recompensas**.



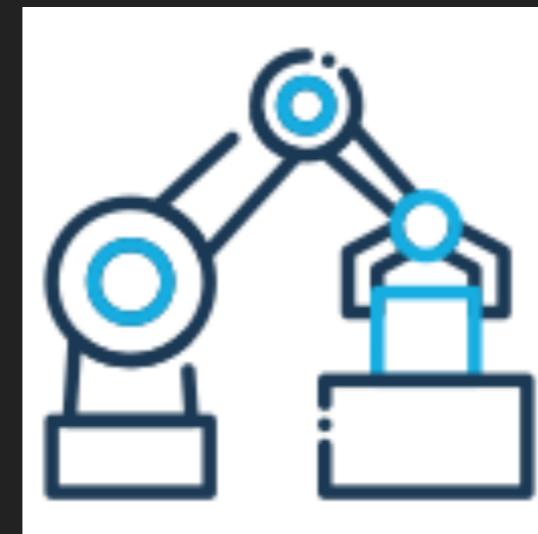
# ¿QUÉ ES APRENDIZAJE POR REFUERZO?

- ▶ El **agente** es una entidad que percibe el **estado** del entorno y toma decisiones (**acciones**) para maximizar una **recompensa** a lo largo del tiempo.
- ▶ El **entorno** es todo lo que rodea al **agente** y con el cual el agente interactúa.
- ▶ El **estado** representa toda la información relevante sobre el **entorno** en un momento. El **agente** necesita conocer el estado para tomar decisiones.
- ▶ Las **acciones** son las decisiones que el **agente** puede tomar en cualquier **estado** dado.
- ▶ La **recompensa** (o **refuerzo**) es un puntaje que le indica al **agente** cuán buena es la **acción** que realizó en relación con el objetivo.



## ¿DE QUÉ MANERA EL AGENTE DECIDE UNA ACCIÓN?

- ▶ Para decidir una determinada **acción** el agente utiliza lo que se conoce como **política** del agente.
- ▶ La **política** es la probabilidad de que el agente elija una determinada acción.
- ▶ La **política** es la estrategia del agente.



# ¿CÓMO SÉ QUE EL AGENTE YA APRENDIÓ?

- ▶ La meta del agente es **maximizar la recompensa acumulada** que recibe en el largo plazo. [Sutton et al., 2018]
- ▶ Si la secuencia de recompensas recibidas después del paso de tiempo  $t$  se denota  $R_{t+1}, R_{t+2}, R_{t+3}, \dots$  y si se busca **maximizar el rendimiento esperado  $G_t$** , entonces  $G_t$  se define como una función específica de la secuencia de recompensas.
- ▶ En el caso más simple, el rendimiento es la suma de las recompensas:

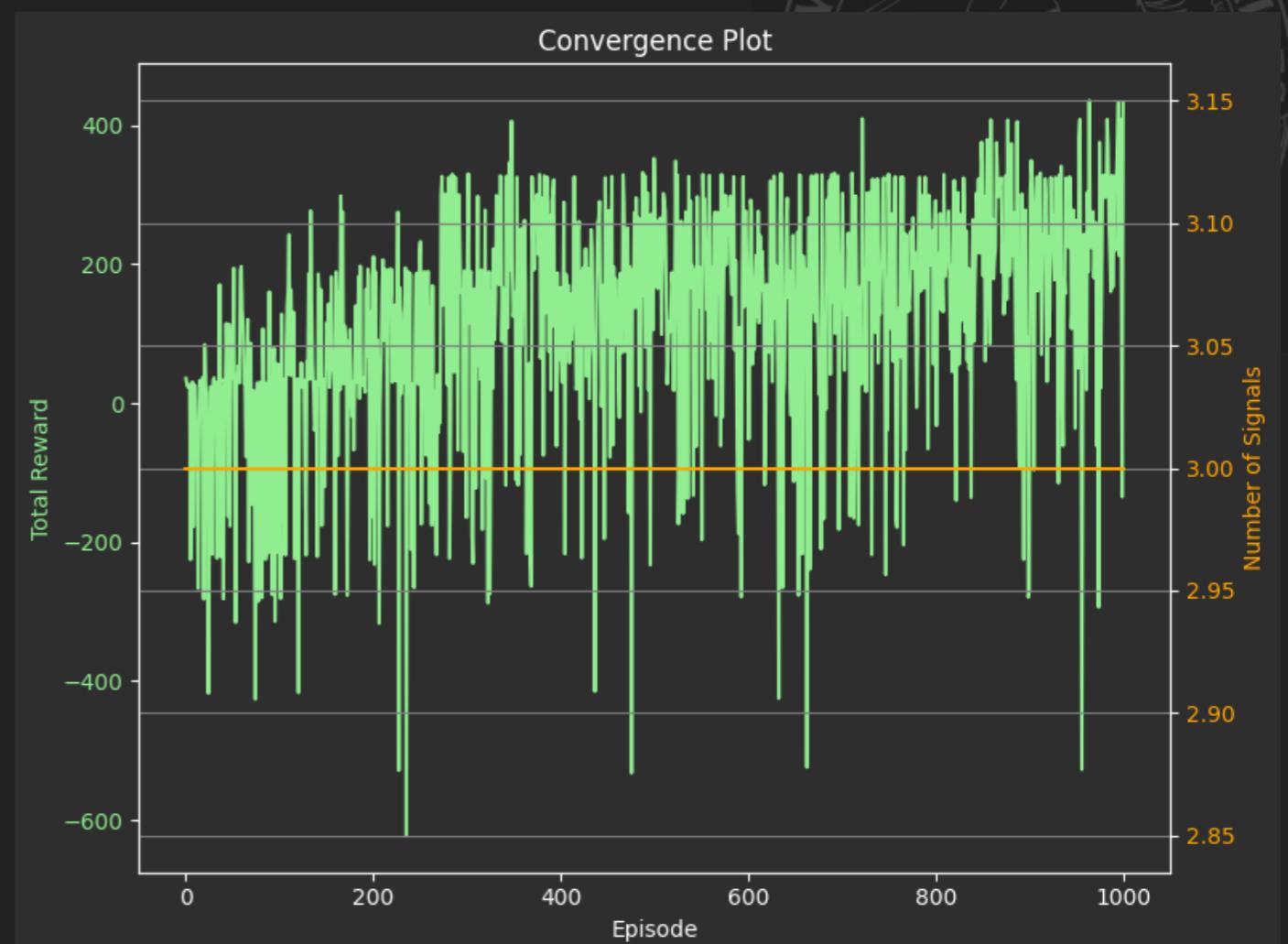
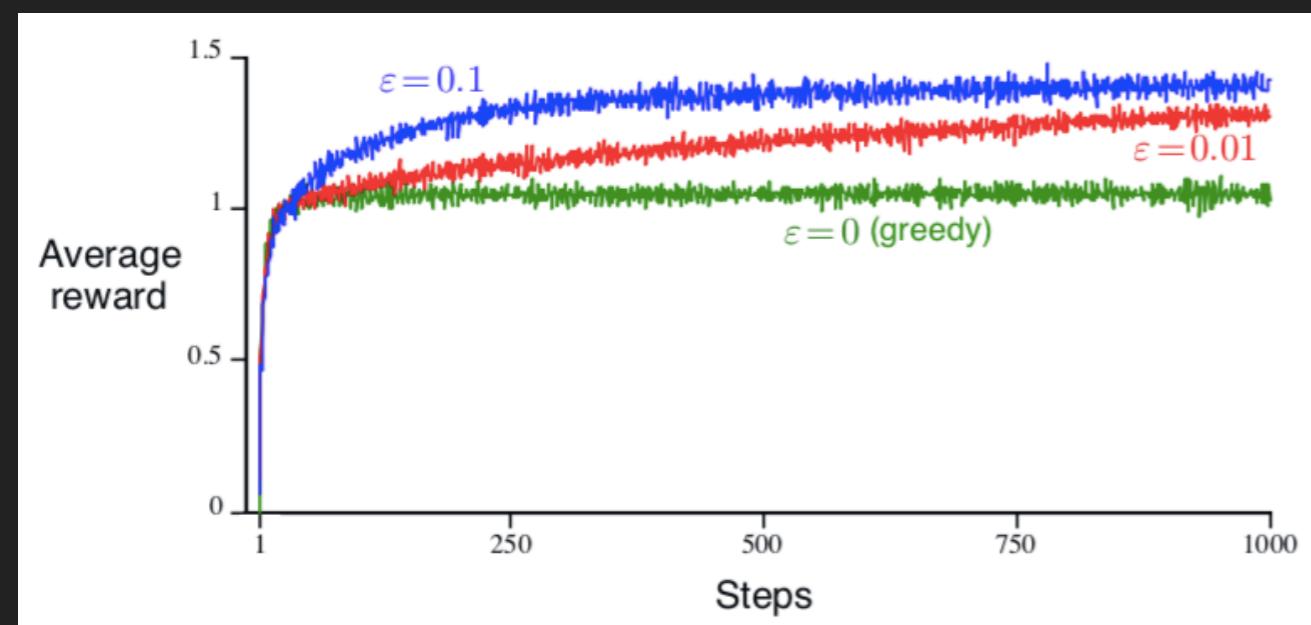
$$G_t \doteq R_{t+1} + R_{t+2} + R_{t+3} + \dots + R_T$$

- ▶ Donde  $T$  es el **paso temporal final**.



# ¿CÓMO SÉ QUE EL AGENTE YA APRENDIÓ?

- ▶ Cuando hay **estabilidad de la recompensa acumulada** (Cumulative Reward)
- ▶ Gráfico de convergencia.



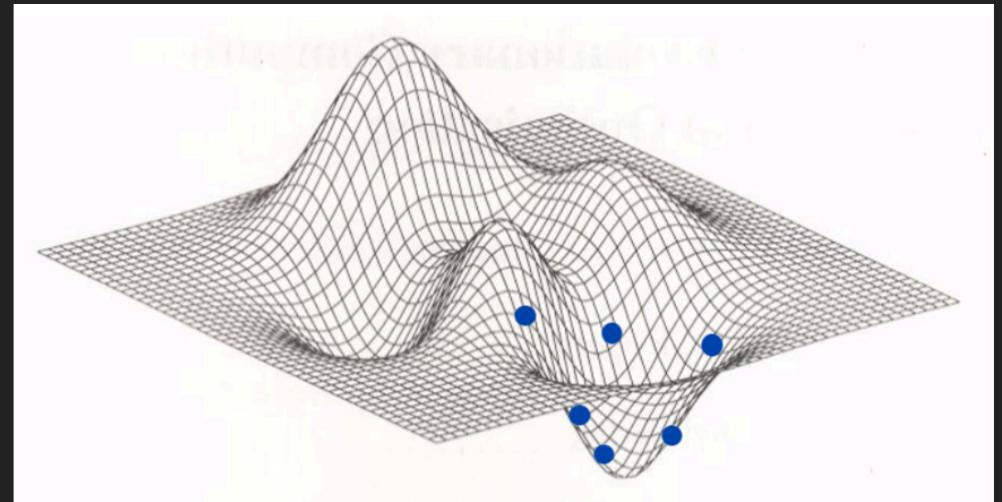
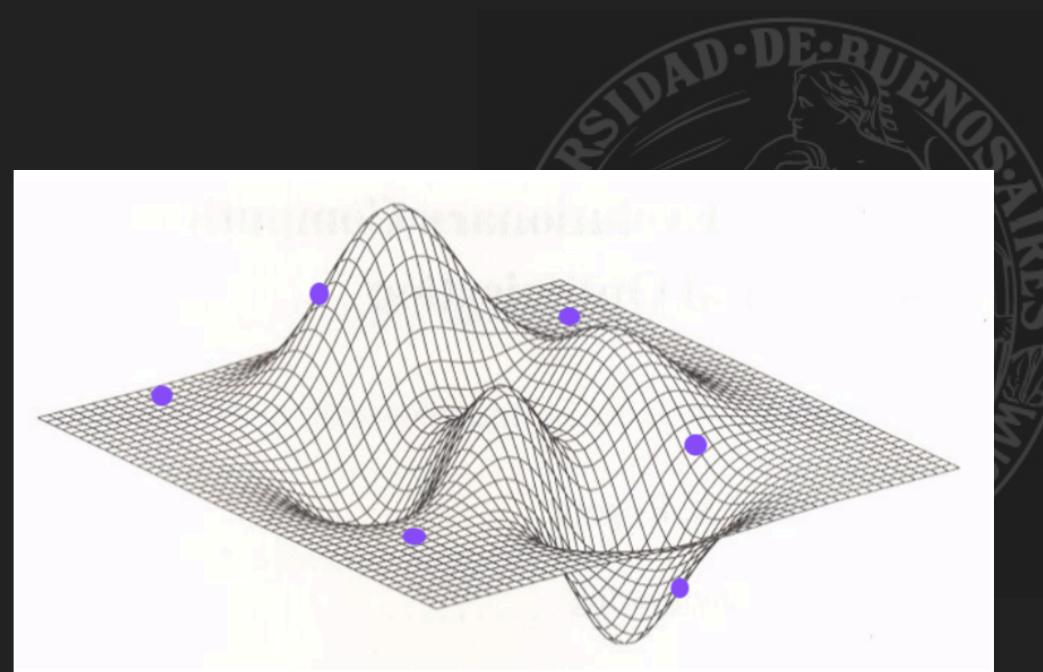
# ¿CÓMO SÉ QUE EL AGENTE YA APRENDIÓ?

- ▶ Cuando la **tasa de cambio en la función de valor** se mantiene aproximadamente constante (Value Function).
- ▶ En algoritmos tales como Q-Learning el valor Q debería estabilizarse.
- ▶ Los valores Q (o cualquier función de valor utilizada) dejan de cambiar significativamente entre episodios.



# ¿CÓMO SÉ QUE EL AGENTE YA APRENDIÓ?

- ▶ Cuando el algoritmo solo **explota lo aprendido**.
- ▶ La **exploración** es el proceso de búsqueda amplia en el espacio de búsqueda para descubrir diferentes regiones que pueden contener soluciones prometedoras.
- ▶ La **explotación** es el proceso de centrarse y mejorar soluciones que ya se sabe que son buenas o prometedoras.



# ¿CÓMO SÉ QUE EL AGENTE YA APRENDIÓ?

- ▶ Cuando se alcanza o supera el **desempeño comparado con una referencia** (Benchmarking).
- ▶ Si el desempeño del agente se aproxima o alcanza un umbral comparado con una solución de referencia o el rendimiento de un agente humano o experto en el entorno, puede ser una señal de que el aprendizaje es suficiente.



# ¿CÓMO SÉ QUE EL AGENTE YA APRENDIÓ?

- ▶ Cuando el **comportamiento en una simulación realista** es la esperada.
- ▶ Si el agente es capaz de realizar sus tareas de manera competente en simulaciones o entornos lo suficientemente realistas y no muestra una disminución en el rendimiento, entonces el agente puede considerarse listo para ser "desplegado" o interrumpir el entrenamiento.
- ▶ Aprendizaje por imitación (IL) (Russell, 1998) o Programming by Demonstration (PbD) (Cypher et al., 1993) o learning from demonstration (LfD) (Schaal, 1997).
- ▶ Behavioral Cloning (BC) (Bain et al., 1999) + Reinforcement Learning (RL).
- ▶ Aprendizaje por Refuerzo Inverso (IRL), (Ng et al., 2000).

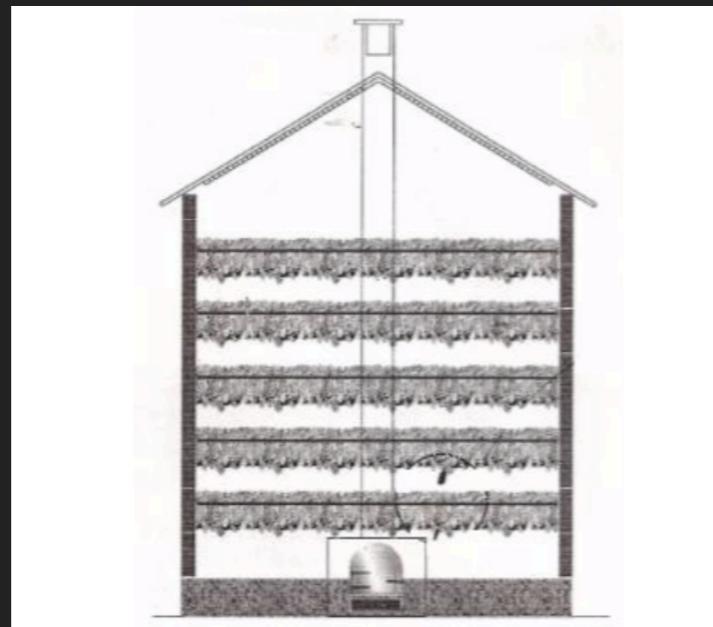


Figura 1. Esquema simplificado de una estufa para secado de tabaco.

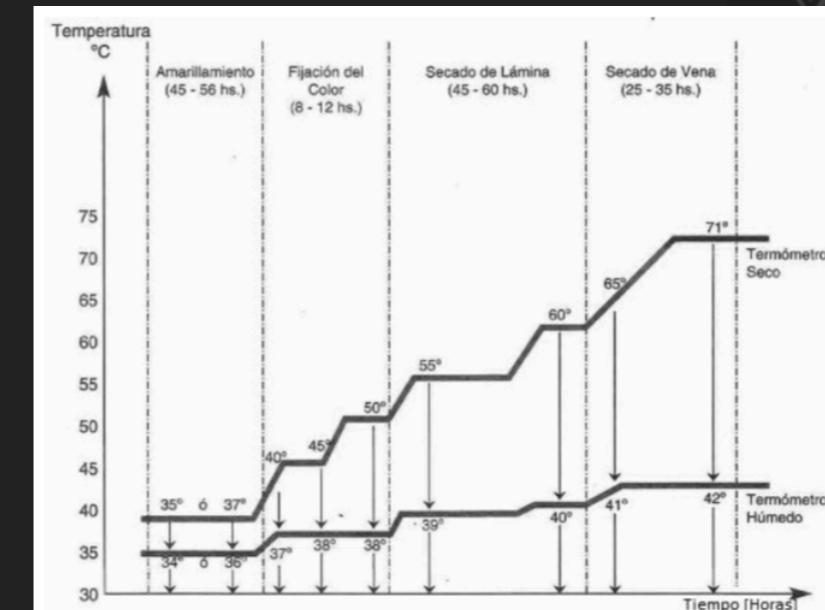


Figura 3. Curvas de temperatura genéricas durante el proceso de secado (Massalín Particulares).



# CUANDO USAR RL?

- ▶ Problemas de toma de decisiones **secuenciales**: es decir, si existen decisiones interdependientes (ajedrez, robot que aprende a caminar).
- ▶ **Entornos dinámicos** o inciertos: si el entorno es cambiante y el agente debe adaptarse en tiempo real (control de tráfico en una ciudad, gestión de inventarios en un almacén, reposición en góndolas).
- ▶ **No hay datos etiquetados**: si  $\exists$  carencia de datasets para entrenar un modelo supervisado pero se puede definir una función de recompensa (entrenar un agente para que aprenda a conducir un automóvil autónomo).
- ▶ Problemas con **recompensas retardadas**: es decir, cuando las acciones del agente tienen consecuencias a cierto plazo (planificación financiera, trading, estrategias de marketing).



## CUANDO NO USAR RL?

- ▶ Problemas **estáticos** o de una sola decisión: el problema no implica una secuencia de decisiones (ej. clasificación de imágenes).
- ▶ **Datos etiquetados disponibles:** si se dispone de datasets etiquetados es mejor usar aprendizaje supervisado (ej. predicción de precios de viviendas).
- ▶ Existen **alternativas más eficientes:** métodos mas simples tales como búsquedas heurísticas o metaheurísticas.



# ¿CÓMO CONSTRUYO EL MODELO MATEMÁTICO DEL AR?

Ejemplo del clima:

- ▶ La probabilidad de que mañana esté **seco** es:
  - ✓ 0.8 si hoy está seco.
  - ✓ 0.6 si hoy llueve.

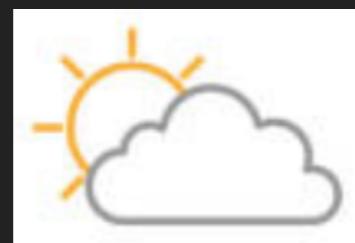


# ¿CÓMO CONSTRUO EL MODELO MATEMÁTICO DEL AR?

- ▶ La evolución del clima para cada día  $t$  es:  $t=0,1,2,\dots$
- ▶ El **estado** del sistema en cada dia  $t$ :
  - ✓ Estado 0 = si el dia  $t$  es seco.
  - ✓ Estado 1 = si el dia  $t$  es lluvioso.
- ▶ Para  $t = 0, 1, 2, \dots$ , la **variable aleatoria**  $X_t$  toma los valores:

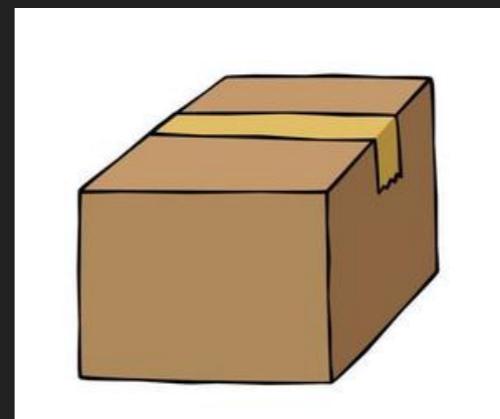
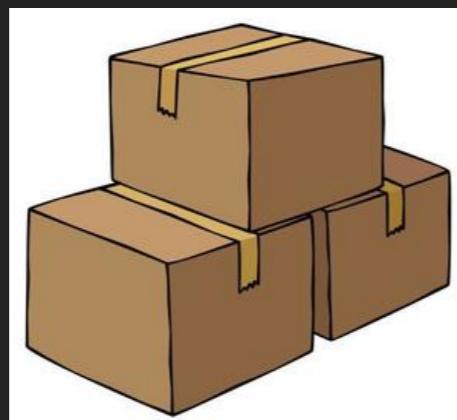
$$X_t = \begin{cases} 0 & \text{si día } t \text{ es seco} \\ 1 & \text{si día } t \text{ es lluvioso} \end{cases}$$

- ▶ La evolución del clima día tras día es un **proceso estocástico**.
- ▶ El **proceso estocástico**  $\{X_t\} = \{X_0, X_1, X_2, \dots\}$  proporciona una representación matemática de la forma en que evoluciona el clima a través del tiempo.



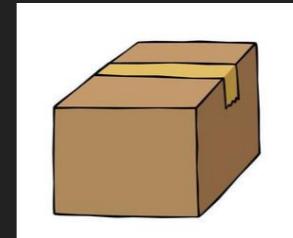
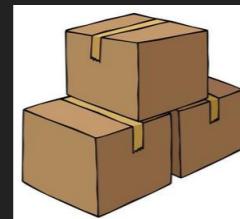
## PROCESOS ESTOCÁSTICOS

- ▶ Un **proceso estocástico** se define como una colección indexada de variables aleatorias  $\{X_t\}$ , donde el índice  $t$  toma valores de un conjunto  $T$  dado.
- ▶  $T$  se considera el conjunto de enteros no negativos mientras que  $X_t$  representa una característica de interés cuantificable en el tiempo  $t$ .
- ▶ Por ejemplo,  $X_t$  puede representar los **niveles de inventario al final de la semana  $t$** .



# PROCESOS ESTOCÁSTICOS

- ▶ Los **procesos estocásticos** describen el comportamiento de un sistema en operación durante algunos periodos.
- ▶ La condición actual del sistema puede estar en una de  $M + 1$  categorías mutuamente excluyentes llamadas **estados**.
- ▶ Por conveniencia en la notación, estos estados se etiquetan  $0, 1, 2, \dots, M$ .
- ▶ La **variable aleatoria  $X_t$**  representa el estado del sistema en el tiempo  $t$ , de manera que sus únicos valores posibles son  $0, 1, \dots, M$ .
- ▶ El **proceso estocástico  $\{X_t\} = \{X_0, X_1, X_2, \dots\}$**  proporcionan una **representación matemática de la forma en que evoluciona la condición del sistema físico a través del tiempo**.
- ▶ Este tipo de procesos se conocen como **procesos estocásticos de tiempo discreto con espacio de estados finito**.



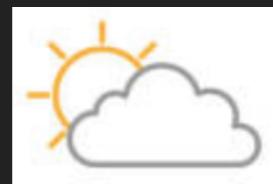
# CADERAS DE MARKOV

$$X_t = \begin{cases} 0 & \text{si día } t \text{ es seco} \\ 1 & \text{si día } t \text{ es lluvioso} \end{cases}$$
$$P\{X_{t+1} = 0 | X_t = 0\} = 0.8,$$
$$P\{X_{t+1} = 0 | X_t = 1\} = 0.6$$



Andrey Markov

- ▶ La probabilidad condicional de cualquier “evento” futuro dados cualquier “evento” pasado y el estado actual  $X_t = i$ , es independiente de los eventos pasados y sólo depende del estado actual del proceso.
- ▶ Se dice que un proceso estocástico  $\{X_t\}$  tiene la propiedad markoviana (es una “cadenas de Markov” o “proceso de Markov”) si  $P\{X_{t+1} = j | X_0 = k_0, X_1 = k_1, \dots, X_{t-1} = k_{t-1}, X_t = i\} = P\{X_{t+1} = j | X_t = i\}$ , para  $t = 0, 1, \dots$  y toda sucesión  $i, j, k_0, k_1, \dots, k_{t-1}$ .



# CADERAS DE MARKOV

$$\begin{aligned} p_{00} &= P\{X_{t+1} = 0 \mid X_t = 0\} = 0.8, \\ p_{10} &= P\{X_{t+1} = 0 \mid X_t = 1\} = 0.6 \end{aligned}$$

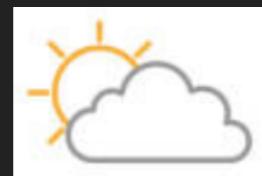
- para todo  $t = 1, 2, \dots$ , éstas probabilidades se conocen como las **probabilidades de transición estacionarias**.
- Las probabilidades condicionales  $P\{X_{t+1} = j \mid X_t = i\}$  de una cadena de Markov se llaman **probabilidades de transición (de un paso)** si para cada  $i$  y  $j$ ,

$$P\{X_{t+1} = j \mid X_t = i\} = P\{X_1 = j \mid X_0 = i\}, \text{ para todo } t=1,2,\dots,$$

- entonces se dice que las probabilidades de transición (de un paso) son **estacionarias**.

$$P\{X_{t+n} = j \mid X_t = i\} = P\{X_n = j \mid X_0 = i\}$$

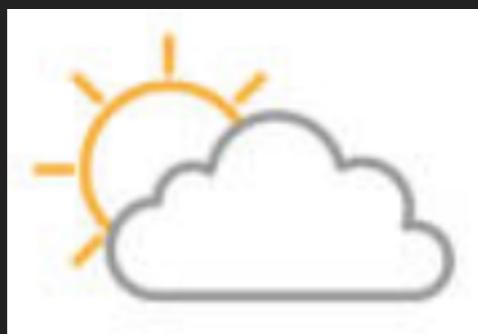
- Si a  $t$  sumamos cualquier entero  $n$  ( $n = 0, 1, 2, \dots$ ) estamos frente a una **probabilidad de transición de  $n$  pasos**.



## CADERAS DE MARKOV

$$\begin{aligned} p_{00} &= P\{X_{t+1} = 0 \mid X_t = 0\} = 0.8, \\ p_{10} &= P\{X_{t+1} = 0 \mid X_t = 1\} = 0.6 \end{aligned}$$

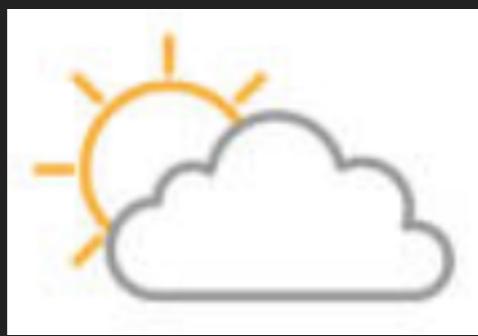
►  $p_{01} = ?$



## CADERAS DE MARKOV

$$\begin{aligned} p_{00} &= P\{X_{t+1} = 0 \mid X_t = 0\} = 0.8, \\ p_{10} &= P\{X_{t+1} = 0 \mid X_t = 1\} = 0.6 \end{aligned}$$

►  $p_{11} = ?$



## CADERAS DE MARKOV

$$\begin{aligned} p_{00} + p_{01} &= 1 \text{ entonces } p_{01} = 1 - 0.8 = 0.2, \\ p_{10} + p_{11} &= 1 \text{ entonces } p_{11} = 1 - 0.6 = 0.4, \end{aligned}$$

La matriz de transición es entonces:

$$\mathbf{P} = \begin{array}{c|cc} \text{Estado} & 0 & 1 \\ \hline 0 & \left[ \begin{matrix} p_{00} & p_{01} \\ p_{10} & p_{11} \end{matrix} \right] \\ 1 & \end{array} = \begin{array}{c|cc} \text{Estado} & 0 & 1 \\ \hline 0 & \left[ \begin{matrix} 0.8 & 0.2 \\ 0.6 & 0.4 \end{matrix} \right] \\ 1 & \end{array}$$

En forma genérica tendremos:

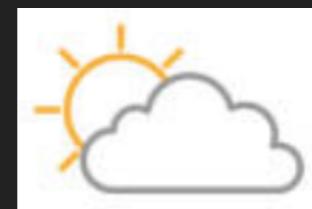
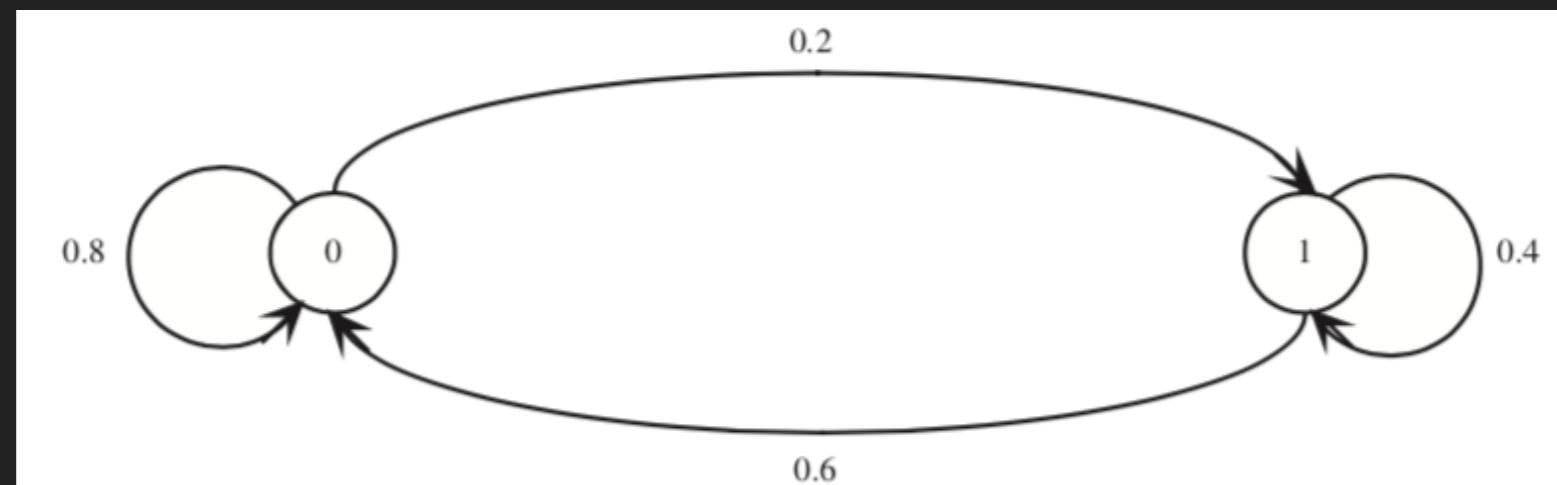
$$\sum_{j=0}^M p_{ij}^{(n)} = 1 \quad \text{para toda } i; n = 0, 1, 2, \dots$$

$$\mathbf{P}^{(n)} = \begin{array}{c|cccc} \text{Estado} & 0 & 1 & \dots & M \\ \hline 0 & \left[ \begin{matrix} p_{00}^{(n)} & p_{01}^{(n)} & \dots & p_{0M}^{(n)} \\ p_{10}^{(n)} & p_{11}^{(n)} & \dots & p_{1M}^{(n)} \\ \vdots & & \dots & \dots \\ p_{M0}^{(n)} & p_{M1}^{(n)} & & p_{MM}^{(n)} \end{matrix} \right] \\ 1 & \\ \vdots & \\ M & \end{array}$$



## CADERAS DE MARKOV

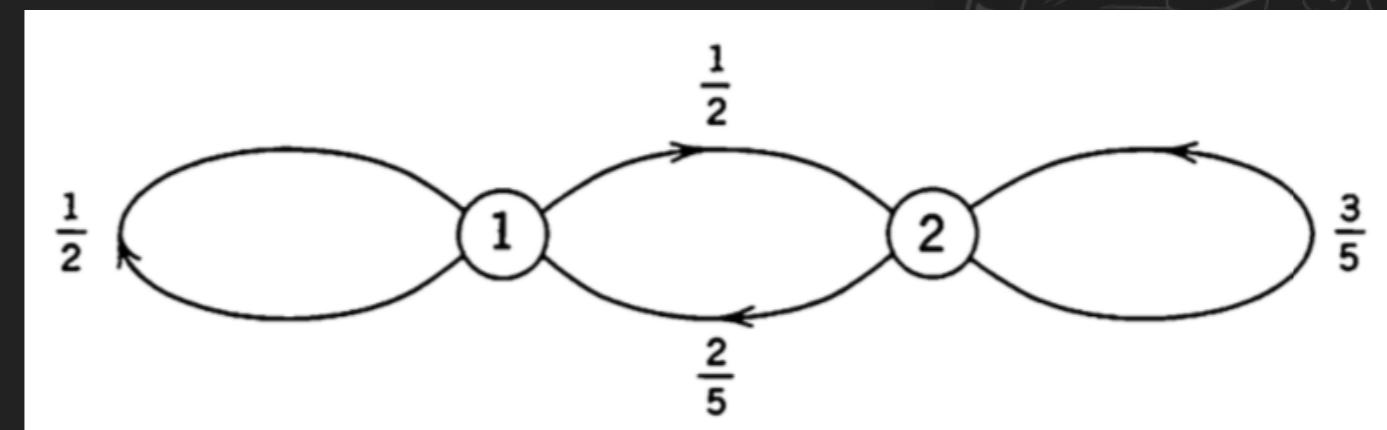
- ▶ Diagrama de transición de estados del ejemplo del clima.



## CADERAS DE MARKOV

- Matriz de transición de estados y diagrama de transición de estados de un fabricante de un producto. (Howard, 1960)

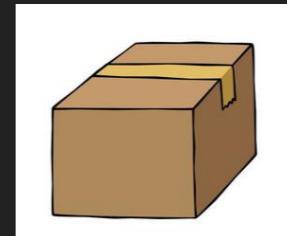
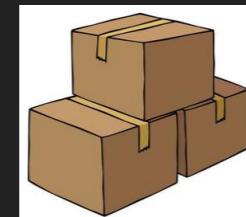
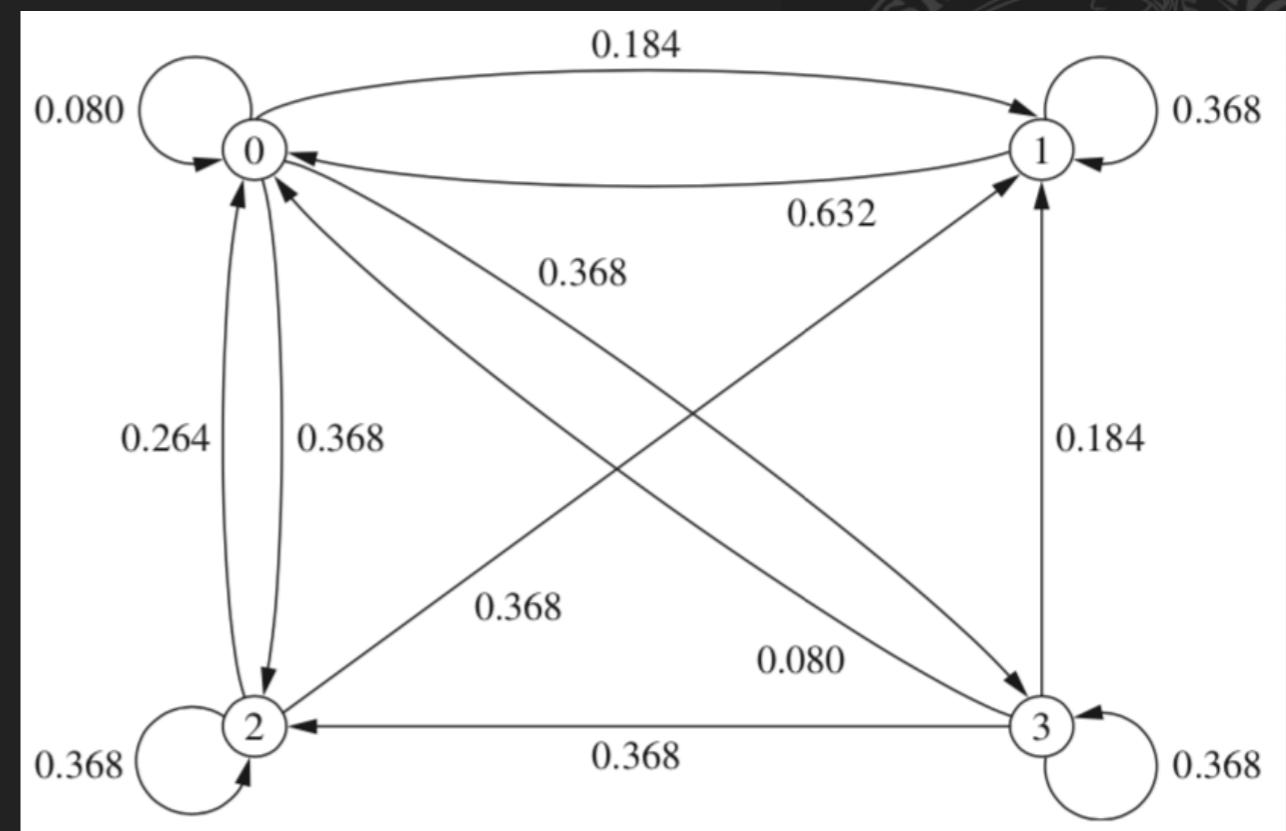
$$\mathbf{P} = [p_{ij}] = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{2}{5} & \frac{3}{5} \end{bmatrix}$$



## CADERAS DE MARKOV

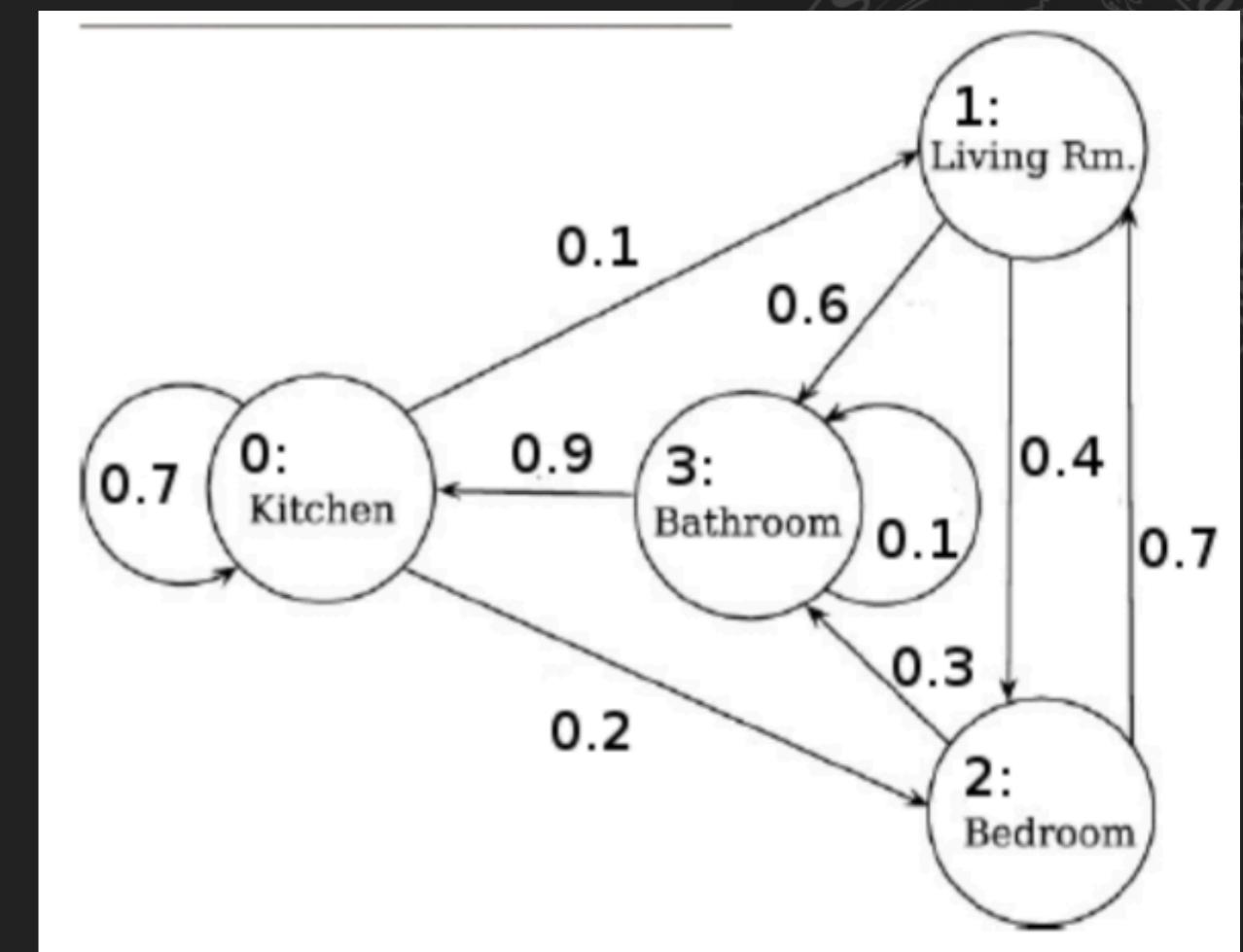
- Matriz de transición de estados y diagrama de transición de estados del ejemplo de inventarios.

$$\mathbf{P} = \begin{array}{c|cccc} \text{Estado} & 0 & 1 & 2 & 3 \\ \hline 0 & 0.080 & 0.184 & 0.368 & 0.368 \\ 1 & 0.632 & 0.368 & 0 & 0 \\ 2 & 0.264 & 0.368 & 0.368 & 0 \\ 3 & 0.080 & 0.184 & 0.368 & 0.368 \end{array}$$



## CADERAS DE MARKOV

- ▶ Diagrama de transición de estados de un robot de limpieza del hogar.



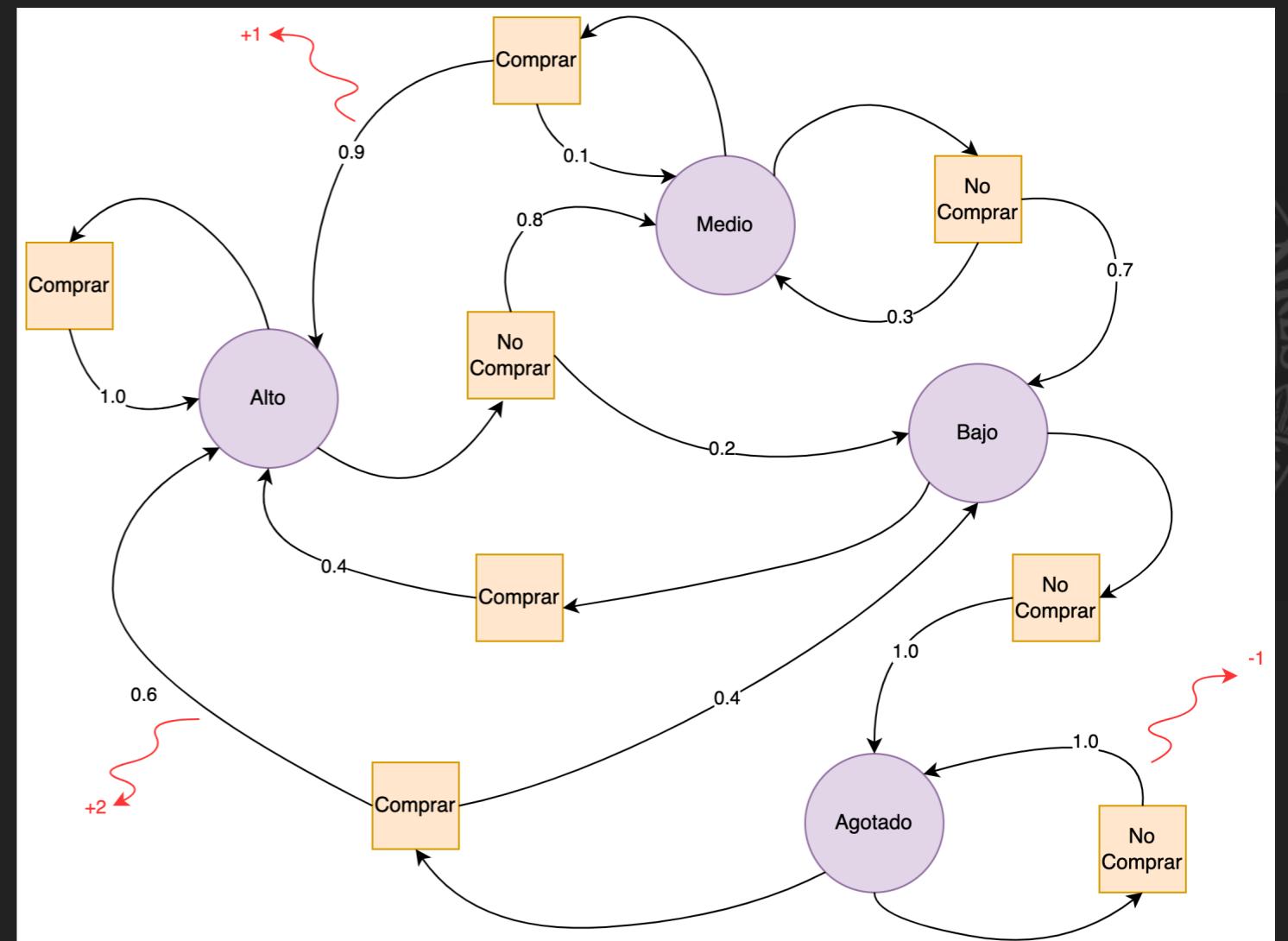
## PREGUNTAS FRECUENTES

- ▶ En el ejemplo de “Inventarios” ¿porque la sumatoria de probabilidades de las columnas no es igual a 1?
- ▶ ¿Porque se dice “cadena” de Markov si no se observa ninguna “cadena” en su representación matricial ni tampoco en su representación gráfica?



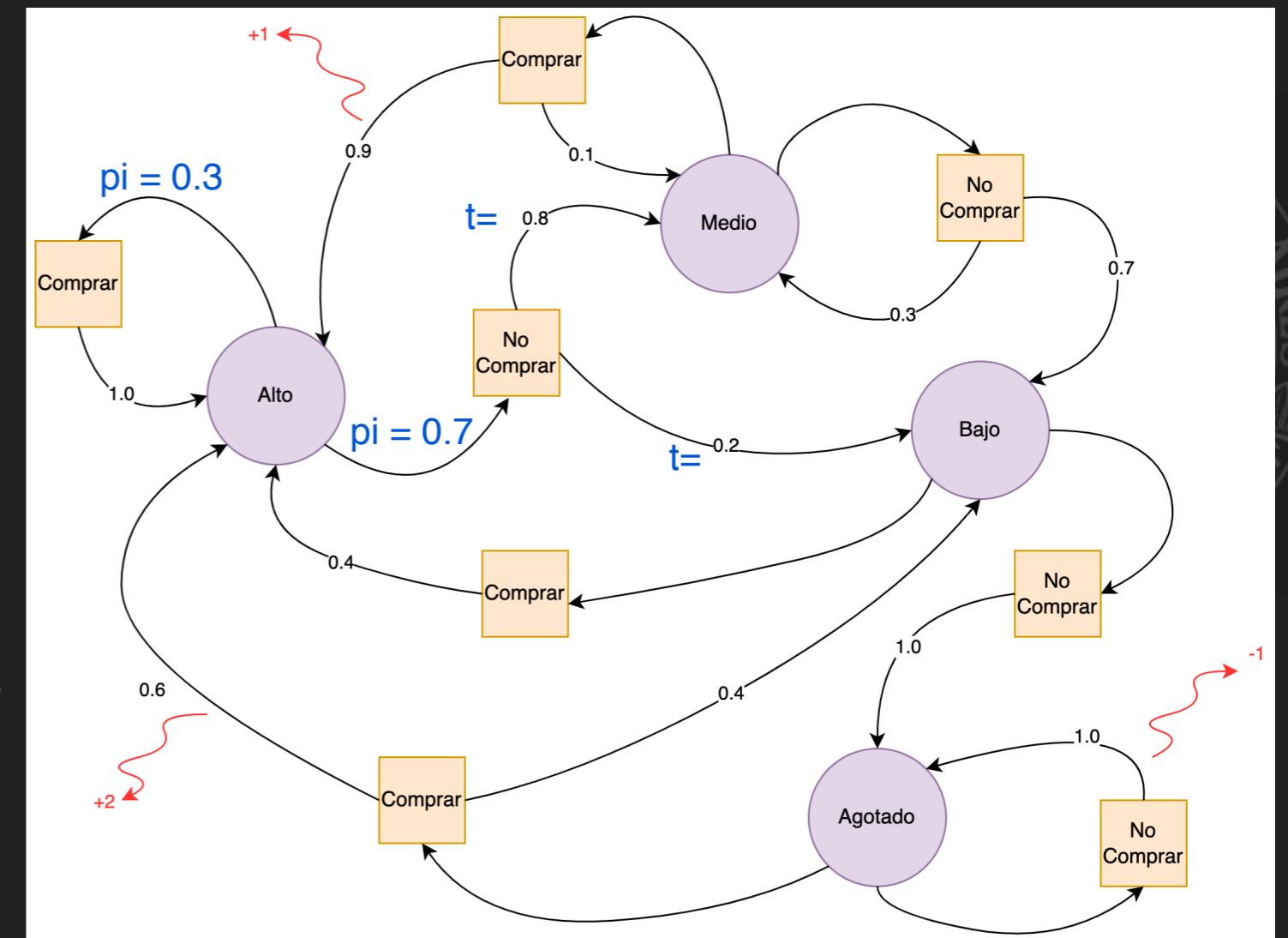
## PROCESOS DE DECISIÓN DE MARKOV (MDP)

- ▶ Supongamos que en una **cadena de Markov** de  $M$  estados se gana cierta cantidad de dinero (o se obtiene un beneficio) cuando se hace una transición del estado  $i$  al estado  $j$ .
- ▶ Maquina de estados finitos
- ▶ El **proceso de Markov** ahora genera una secuencia de recompensas a medida que hace transiciones de estado a estado.



## PROCESOS DE DECISIÓN DE MARKOV (MDP)

- ▶ Un proceso de decisión de Markov es un proceso de control estocástico de tiempo discreto donde los estados son en parte aleatorios y en parte bajo el control de un tomador de decisiones (decision maker).
- ▶ El DM es un agente que toma acciones en cada estado.

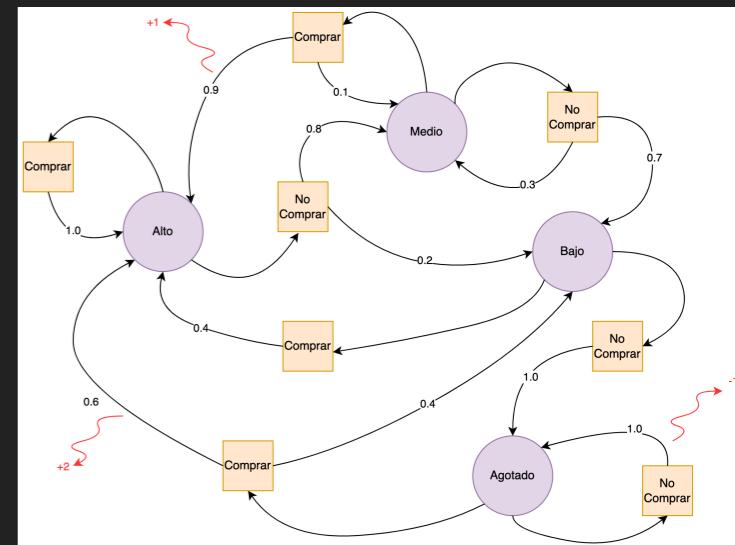


# MODELO MATEMÁTICO DE UN MDP

► Formalmente, un **MDP** es una tupla  $M = \langle S, A, \Phi, R \rangle$  (Puterman, 1994)

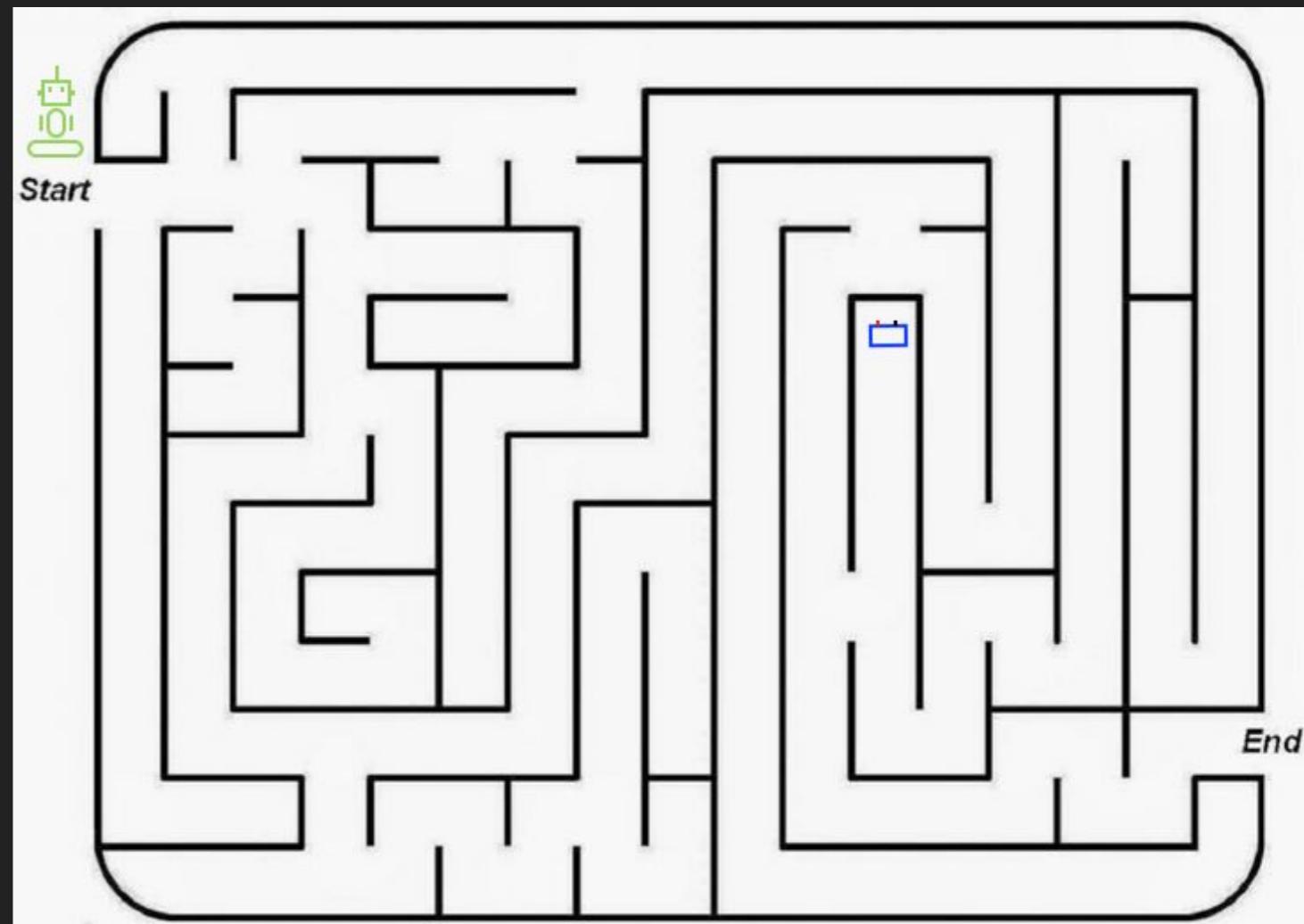
donde existe:

- ✓ Un conjunto finito de estados  $S : \{s_1, \dots, s_n\}$ , donde  $s_t$  denota el estado  $s \in S$  al tiempo  $t$ .
- ✓ Un conjunto finito de acciones que pueden depender de cada estado,  $A(s)$ , donde  $a_t \in A(s)$  denota la acción realizada en un estado  $s$  en el tiempo  $t$ .
- ✓ Una función de recompensa ( $R^{a_{ss'}}$ ) que devuelve un número real indicando lo deseado de estar en un estado  $s' \in S$  dado que en el estado  $s \in S$  se realizó la acción  $a \in A(s)$ . La recompensa  $R(s,a)$  es el valor que el agente recibe por realizar una acción  $a$  en el estado  $s$ . Si  $R$  es positivo (**recompensa**) o negativo o cero (**penalización**).
- ✓ Una función de transición de estados dada como una distribución de probabilidad ( $P^{a_{ss'}}$ ) que denota la probabilidad de llegar al estado  $s' \in S$  dado que se tomó la acción  $a \in A(s)$  en el estado  $s \in S$ , que también denotaremos como  $\Phi(s, a, s')$ .



## RECOMPENSA

- ▶ Supongamos que un robot busca una batería en un laberinto.
- ▶ Las recompensas recibidas después de un tiempo  $t$  se denotan como:  $r_{t+1}, r_{t+2}, r_{t+3}, \dots$ , lo que se quiere es **maximizar la recompensa total recibida**.



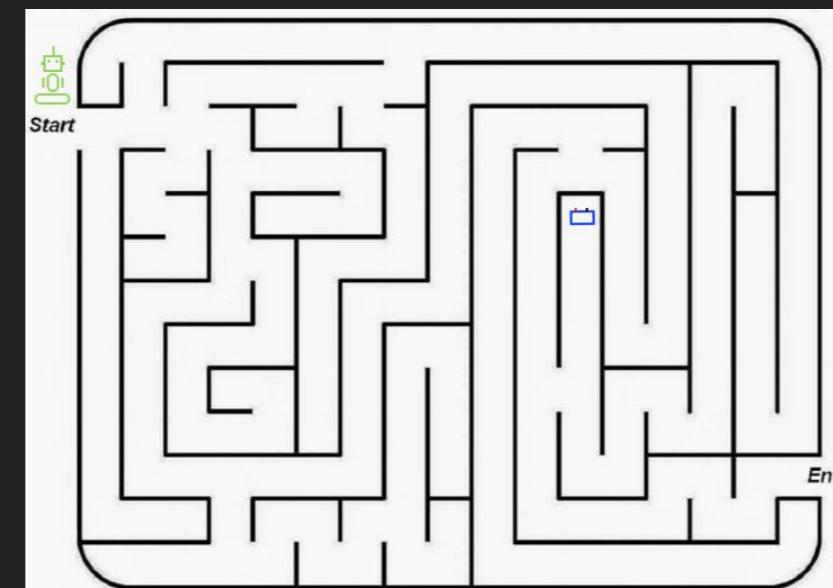
# RECOMPENSA

- ▶ Podemos calcular la recompensa  $R_t$  de 2 formas:

$$R_t = r_{t+1} + r_{t+2} + r_{t+3} + \dots + r_T$$

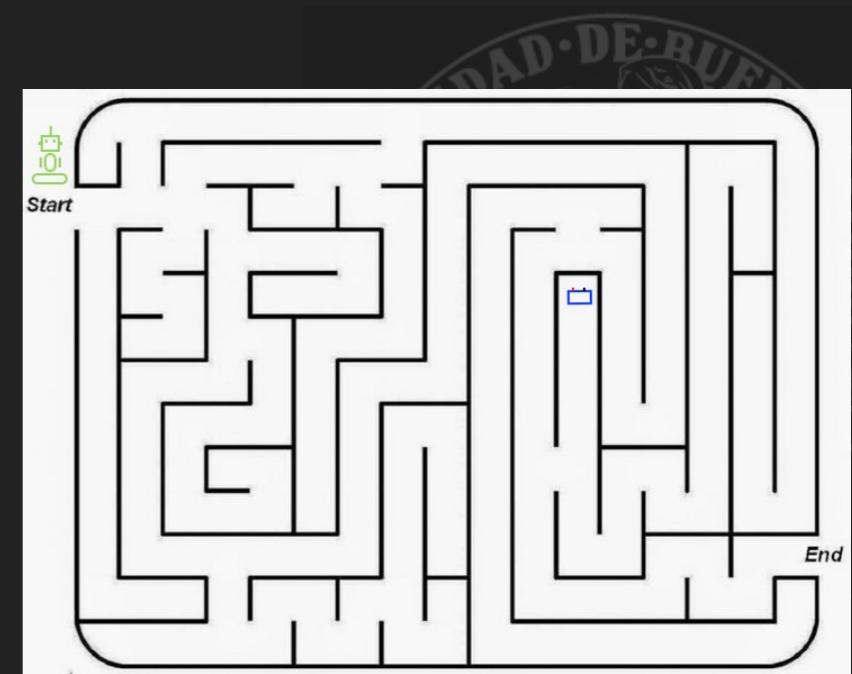
$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \dots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$$

- ▶ donde  $\gamma$  es la tasa de descuento, siendo  $\gamma \in [0, 1]$



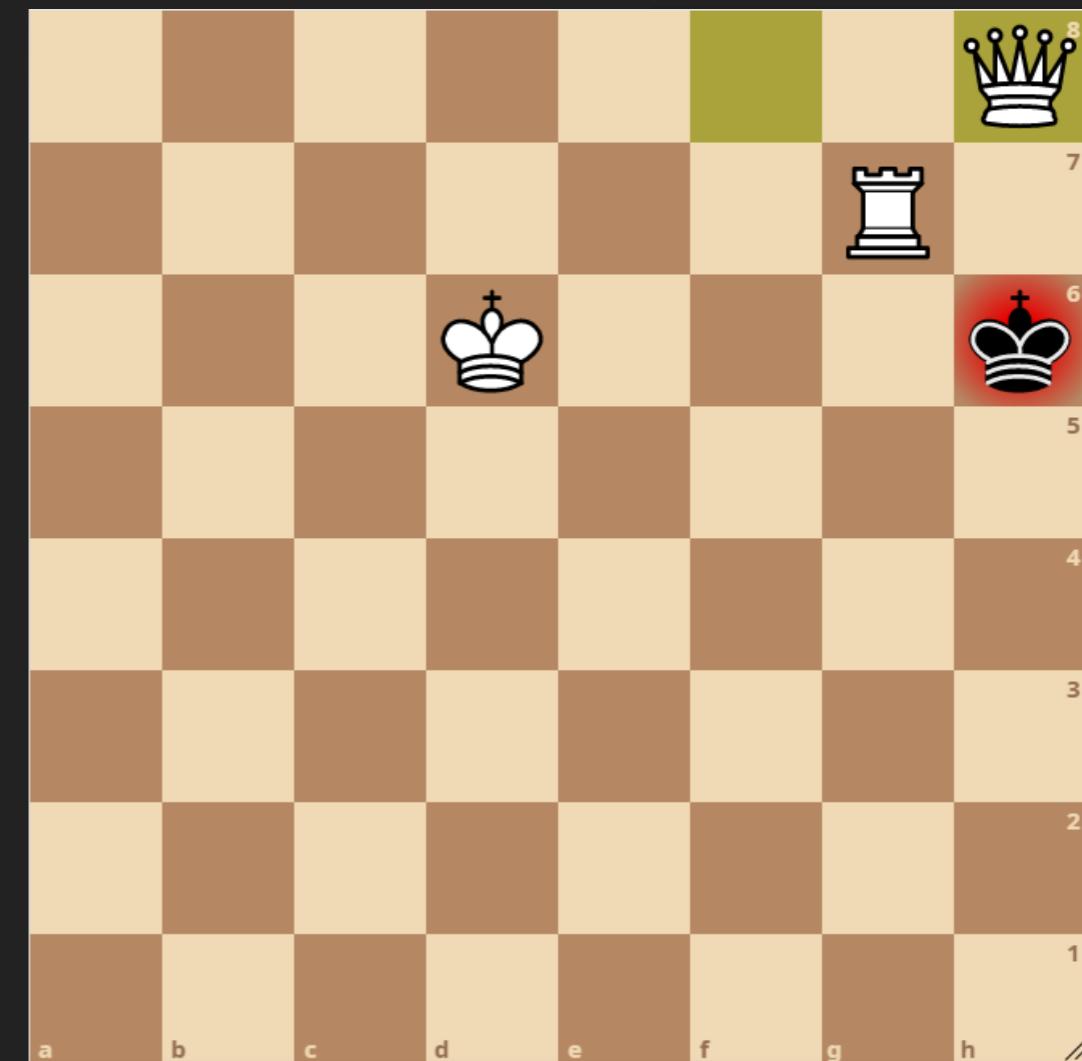
# RECOMPENSA

- ▶ Si el robot no usa tasa de descuento ( $\gamma=1$ ), podría aprender rutas más largas porque **todas las recompensas futuras valen lo mismo**.
- ▶ Si el robot usa tasa de descuento ( $0<=\gamma<=1$ ), aprenderá a llegar más rápido, ya que **las recompensas lejanas valen menos**.
- ▶  $\gamma$  ayuda a maximizar la recompensa de manera más eficiente.
- ▶ Si no usáramos **tasa de descuento**, el robot podría:
  - ✓ Perder tiempo en rutas largas.
  - ✓ Quedarse atrapado en bucles infinitos si el problema no tiene un final.
- ▶ La **tasa de descuento** balancea la recompensa inmediata con la futura, favoreciendo decisiones óptimas en el tiempo.
- ▶ La **tasa de descuento  $\gamma$**  es un factor que determina cuánta importancia se le da a las recompensas futuras en comparación con las recompensas inmediatas.



# TAREAS EPISÓDICAS Y TAREAS CONTINUAS

- ▶ Ejemplo de Frozen Lake y de Ajedrez



## TAREAS EPISÓDICAS Y TAREAS CONTINUAS

- ▶ En Frozen Lake el personaje realiza una **tarea episódica** porque el episodio termina al caer al lago (por debilidad del hielo) o llegar a la meta.
- ▶ En el ajedrez la partida es una **tarea episódica** porque finaliza cuando un jugador gana, hace tabla o abandona.



## TAREAS EPISÓDICAS Y TAREAS CONTINUAS

- ▶ La tarea de un agente (o bot) que hace trading en la bolsa de comercio es una **tarea continua**.



# TAREAS EPISÓDICAS Y TAREAS CONTINUAS

- ▶ Si el agente sigue funcionando para siempre (**tarea continua**), la suma de las recompensas sería infinita, lo cual es un problema matemático.

- ▶ Ejemplo sin tasa de descuento:

$$R_t = 10 + 10 + 10 + 10 + 10 + \dots (\infty)$$

- ▶ Para evitar esto, aplicamos una tasa de descuento  $\gamma$  ( $0 \leq \gamma \leq 1$ ).
- ▶ En vez de sumar todas las recompensas con el mismo peso, hacemos que las recompensas futuras valgan menos:

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \dots$$

- ▶ Si  $\gamma=0.9$ , entonces:

$$\begin{aligned} R_t &= 10 + 0.9 \cdot 10 + 0.9^2 \cdot 10 + 0.9^3 \cdot 10 + \dots \\ &= 10 + 9 + 8.1 + 7.29 + \dots \end{aligned}$$

- ▶ En este caso, la suma ya no se va al infinito porque **cada término es cada vez más pequeño**.



# TAREAS EPISÓDICAS Y TAREAS CONTINUAS

- ▶ **Tareas episódicas:** Tienen un final claro (un episodio). Ejemplo: un juego que termina cuando el jugador gana.
- ▶ **Tareas continuas:** No tienen un final definido. Ejemplo: un robot en una fábrica que nunca deja de operar.

Si las recompensas recibidas después de un tiempo  $t$  se denotan como:  $r_{t+1}, r_{t+2}, r_{t+3}, \dots$ , lo que se quiere es maximizar la recompensa total ( $R_t$ ), que en el caso más simple es:

$$R_t = r_{t+1} + r_{t+2} + r_{t+3} + \dots + r_T.$$

Si se tiene un punto terminal se llaman tareas *episódicas*, si no se tiene se llaman tareas *continuas*. En este último caso, la fórmula de arriba presenta problemas, ya que no podemos hacer el cálculo cuando  $T$  no tiene límite. Podemos usar una forma alternativa en donde se van haciendo cada vez más pequeñas las contribuciones de las recompensas más lejanas:

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \dots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1},$$

donde  $\gamma$  se conoce como la *razón de descuento* y está entre:  $0 \leq \gamma < 1$ . Si  $\gamma = 0$  se trata de maximizar la recompensa total tomando en cuenta sólo las recompensas inmediatas.

## ESTADOS ABSORBENTES

- ▶ En Frozen Lake si el personaje cae al lago o llega a la meta entra en un **estado absorbente**.
- ▶ Un **estado absorbente** es un estado final en el que, una vez que el agente entra, no puede salir y las recompensas futuras se mantienen constantes (generalmente 0).
- ▶ Si el agente llega al estado  $s_{abs}$ , entonces:
  - ✓ Cualquier acción lo mantiene en  $s_{abs}$  (es decir,  $P(s_{abs}|s_{abs}, a) = 1$ ).
  - ✓ Las recompensas después de ese estado son 0 o constantes ( $R(s_{abs}, a) = 0$  o alguna otra cantidad fija).



## VALOR DE UN ESTADO

- ▶ Si lanzamos un dado de seis caras, el **valor esperado** del resultado es:

$$\mathbb{E}[X] = 1 \cdot \frac{1}{6} + 2 \cdot \frac{1}{6} + 3 \cdot \frac{1}{6} + 4 \cdot \frac{1}{6} + 5 \cdot \frac{1}{6} + 6 \cdot \frac{1}{6} = 3.5$$

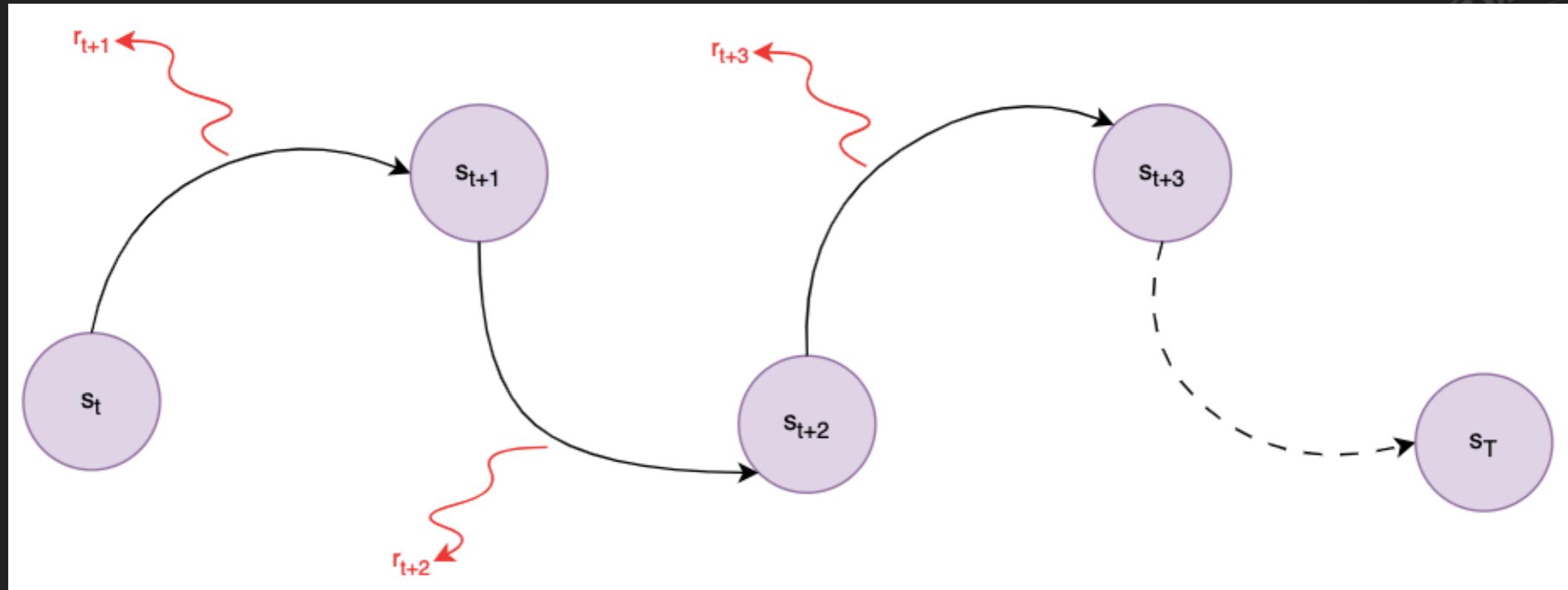


- ▶ Esto significa que, en promedio, si lanzamos el dado muchas veces, el **valor esperado** del resultado será 3.5.
- ▶ **Esperanza matemática:** es un valor que indica el resultado promedio que se espera obtener de una variable aleatoria.

$$\mathbb{E}[X] = \sum_i x_i P(x_i)$$

## VALOR DE UN ESTADO

- ▶ ¿Cómo sé “cuán bueno” es un estado?



## VALOR DE UN ESTADO

- La manera más simple de medir el valor de un estado de un MDP es obtener la recompensa inmediata que recibe el agente por moverse al estado siguiente  $s_{t+1}$ .

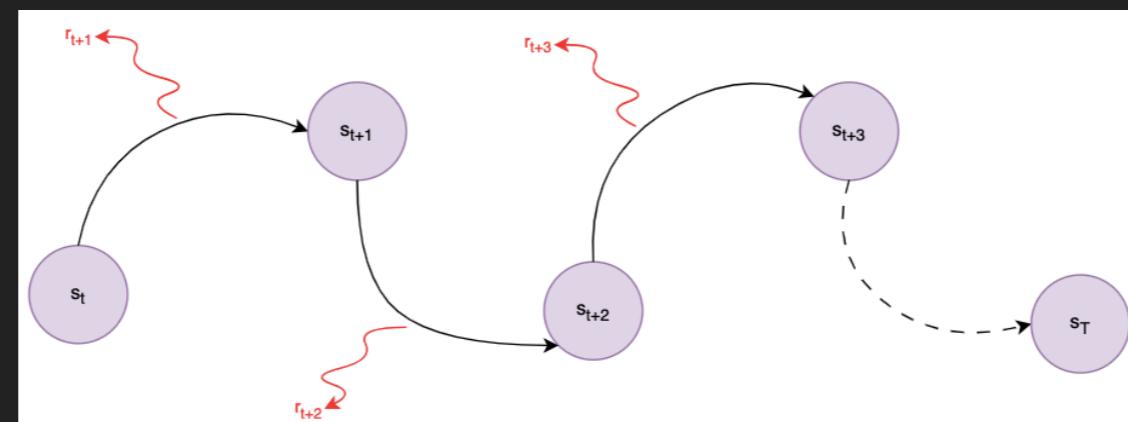
$$V(s_t) = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \dots + r_T$$

$$V(s_{t+1}) = r_{t+2} + \gamma r_{t+3} + \gamma^2 r_{t+4} + \dots + r_T$$

- Identificamos el patrón recursivo. El término  $\gamma r_{t+2} + \gamma^2 r_{t+3} + \dots$  es muy similar a la función de valor del siguiente estado  $s_{t+1}$
- Podemos sustituir  $V(s_{t+1})$  en la expresión  $V(s_t)$

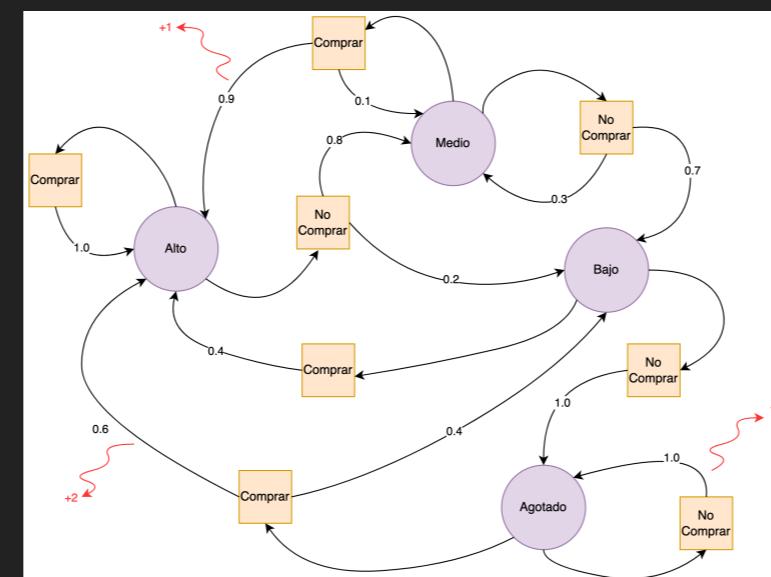
$$V(s_t) = r_{t+1} + \gamma * [r_{t+2} + \gamma r_{t+3} + \dots + r_T]$$

$$V(s_t) = r_{t+1} + \gamma * V(s_{t+1})$$

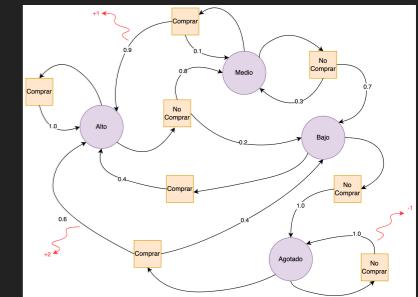


## VALOR DE UN ESTADO

- ▶ Si el agente se encuentra en un **estado  $s$** , la **política  $\pi$**  es la estrategia o regla de cual **acción** decide tomar el agente desde ese estado  $s$ .
- ▶ Una política podría ser elegir la mejor acción posible desde un estado  $s$ .



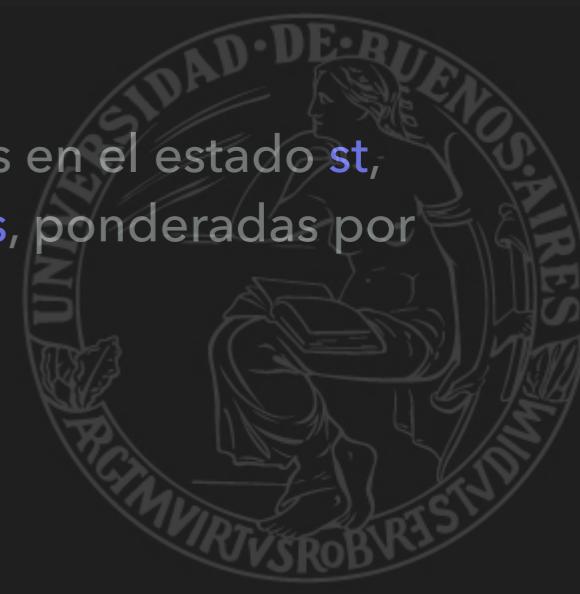
# VALOR DE UN ESTADO



- ▶ Dado que estamos en un entorno incierto o de incertidumbre (porque no podemos predecir el estado siguiente debido a la existencia de probabilidades) el **valor de un estado** se puede calcular a partir del **valor esperado (esperanza)** por emplear una **política  $\pi$** .
- ▶ La **esperanza** representa el valor promedio que podemos esperar obtener si estamos en el estado  $s_t$ , seguimos la **política  $\pi$** , y consideramos todas las **posibles transiciones y recompensas**, ponderadas por sus probabilidades.

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

- ▶ El **valor de estar en el estado  $s_t$  siguiendo la **política  $\pi$****  es igual al **valor esperado** de la **recompensa inmediata  $r_t$**  mas el **valor descontado** del siguiente estado  $s_{t+1}$ , también evaluado siguiendo la **política  $\pi$** .
- ▶ El **valor esperado** bajo la **política  $\pi$**  implica promediar sobre todas las posibles acciones que la **política  $\pi$**  podría recomendar en ese **estado**, y sobre todos los posibles **estados siguientes** que se podría alcanzar al tomar esas **acciones**.
- ▶ Esto se conoce como **Ecuación de Bellman para la evaluación de una política en un estado  $s_t$** .



## EJEMPLO

- ▶ Supongamos que queremos modelar el comportamiento de un **robot** que repone mercadería en un supermercado.
- ▶ El **objetivo** es determinar cuál es el valor de cada **estado** dada una política específica.



## EJEMPLO

► La góndola tiene 3 estados posibles:

- ✓ **Alto**: La góndola está llena de productos.
- ✓ **Medio**: La góndola tiene una cantidad moderada de productos.
- ✓ **Bajo**: La góndola está casi vacía.

► El robot tiene 2 acciones posibles:

- ✓ **Reponer**: El robot repone la góndola, llevándola al estado "Alto".
- ✓ **No Reponer**: El robot no hace nada.



## EJEMPLO



- ▶ Las probabilidades de transición del MDP se expresan en forma tabular:

Estado Actual	Acción	Estado Siguiente	Probabilidad
Alto	Reponer	Alto	1
Alto	No Reponer	Medio	0.7
Alto	No Reponer	Bajo	0.3
Medio	Reponer	Alto	1
Medio	No Reponer	Bajo	0.8
Medio	No Reponer	Medio	0.2
Bajo	Reponer	Alto	1
Bajo	No Reponer	Agotado	0.9
Bajo	No Reponer	Bajo	0.1



## EJEMPLO



- ▶ Las recompensas del MDP son:

Estado Actual	Acción	Recompensa
Alto	Reponer	5
Alto	No Reponer	8
Medio	Reponer	2
Medio	No Reponer	4
Bajo	Reponer	1
Bajo	No Reponer	0



## EJEMPLO



► La política  $\pi$  (el comportamiento del robot) se define como:

- ✓ Si el estado es "Alto", el robot nunca repone.
- ✓ Si el estado es "Medio", el robot a veces repone (con probabilidad 0.5).
- ✓ Si el estado es "Bajo", el robot siempre repone.

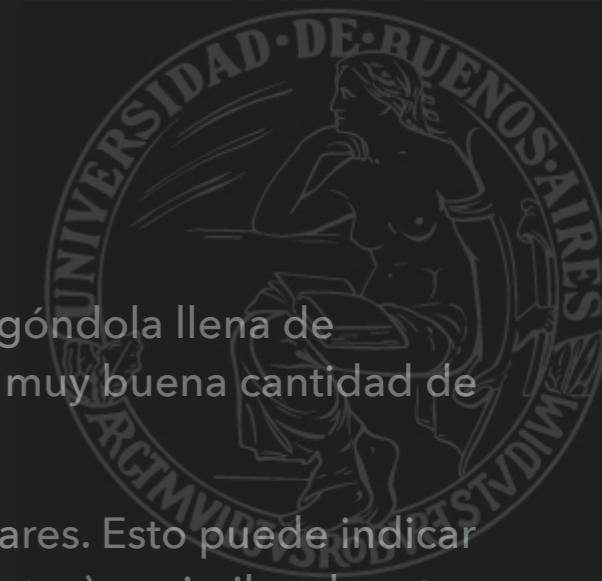
Estado	Acción "Reponer"	Acción "No Reponer"
Alto	0	1
Medio	0.5	0.5
Bajo	1	0



## CONCLUSION DEL EJEMPLO



- ▶ El robot nunca repone en "Alto", a veces repone en "Medio", y siempre repone en "Bajo"), el valor de cada estado (es decir, la recompensa total esperada a largo plazo) es:
  - ✓ Alto: 47.90 (aproximadamente)
  - ✓ Medio: 44.44 (aproximadamente)
  - ✓ Bajo: 44.11 (aproximadamente)
- ▶ **Alto tiene el valor más alto:** Esto indica que, a largo plazo, es mejor estar en el estado "Alto" (con la góndola llena de productos) que en los estados "Medio" o "Bajo". Eso de lógico, ya que en ese estado se genera una muy buena cantidad de ventas.
- ▶ **La diferencia entre Medio y Bajo es pequeña:** Los estados "Medio" y "Bajo" tienen valores muy similares. Esto puede indicar que el costo de pasar del estado "Medio" al estado "Bajo" (perder ventas debido a la falta de productos) es similar al costo de siempre reponer cuando se está en el estado "Bajo".
- ▶ **Relación con la Política:** Es importante recordar que estos valores son específicos para la política que definimos. Si cambiáramos la política (por ejemplo, si el robot siempre repone en "Medio"), los valores de los estados también cambiarían.
- ▶ **No es la Política Óptima:** Estos valores no nos dicen cuál es la mejor política. Solo nos dicen **qué tan buena es la política que estamos evaluando**. Para encontrar la mejor política, necesitaríamos probar diferentes políticas y comparar sus valores.



# REFERENCIAS BIBLIOGRÁFICAS Y WEB (I)

- ▶ S. Wang, S. Zhang, J. Zhang, R. Hu, X. Li, T. Zhang, et al., "Reinforcement Learning Enhanced LLMs: A Survey," arXiv preprint arXiv:2412.10400, 2024.
- ▶ J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal Policy Optimization Algorithms," arXiv preprint arXiv:1707.06347, 2017.
- ▶ Hillier, F. S. Lieberman, G. J. (2010). Introducción a la Investigación de Operaciones.
- ▶ D. Guo et al., "DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning," arXiv preprint arXiv: 2501.12948, 2025.
- ▶ R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction, 2nd ed. Cambridge, MA, USA: MIT Press, 2018.
- ▶ Cypher, A., & Halbert, D. C. (Eds.). (1993). Watch what I do: programming by demonstration. MIT press. ISO 690.
- ▶ Bain, M. and Sommut, C., 1999. A framework for behavioural cloning. Machine Intelligence 15, 15:103.
- ▶ Schaal, S. Learning from demonstration. In Advances in neural information processing systems, pages 1040-1046, 1997.
- ▶ S. Russell, "Learning agents for uncertain environments (extended abstract)," Proceedings of the 11th Annual Conference on Computational Learning Theory (COLT'98), Madison, WI, USA, 1998, pp. 101-103. doi: 10.1145/279943.279964.
- ▶ Ng, A. Y., & Russell, S. (2000). "Algorithms for inverse reinforcement learning." Proceedings of the Seventeenth International Conference on Machine Learning (ICML), vol. 1, no. 2, p. 2.



## REFERENCIAS BIBLIOGRÁFICAS Y WEB (II)

- ▶ <https://www.rubblemaster.com/es/informes-de-aplicacion/roca-natural/preparacion-de-la-piedra-caliza/gruenerer-zement-mit-rm-100go/>
- ▶ Howard, R. A. (1960). Dynamic programming and Markov processes.
- ▶ Puterman, M. L.: Markov Decision Processes: Discrete Stochastic Dynamic Programming. Wiley (1994).
- ▶ Bellman, R. E.: Dynamic Programming. Princeton University Press (1957) .

