

# APRENDIZAJE REFORZADO

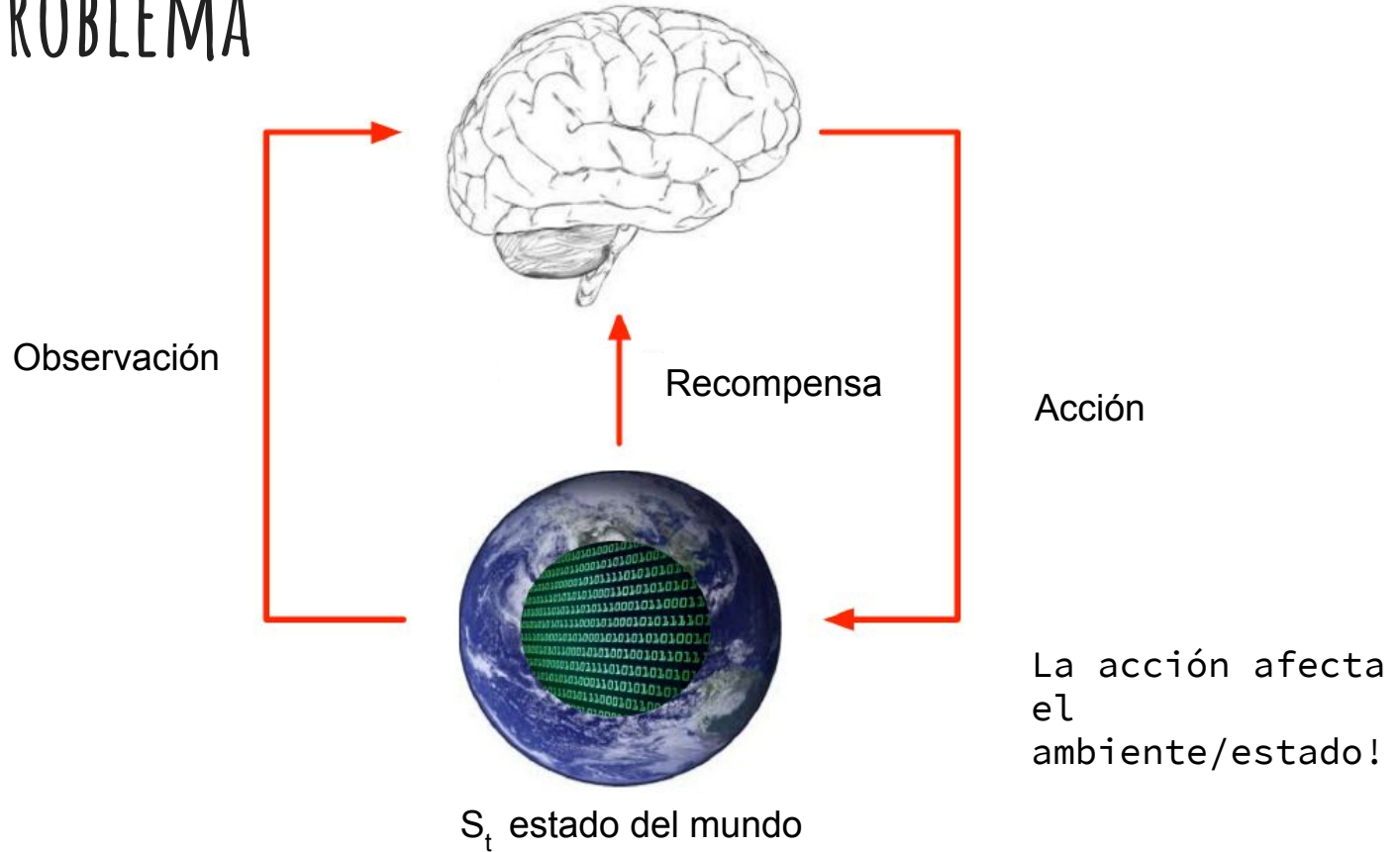
## CLASE 1

**Julían Martínez**

EL CONTENIDO DE ESTE  
CURSO FUE DESARROLLADO  
EN GRAN PARTE CON LA  
AYUDA DE JAVIER KREINER



# MARCO DEL PROBLEMA





# DIFERENCIAS CON OTROS PARADIGMAS DE ML, RL VS APRENDIZAJE SUPERVISADO

- No viene dado un dataset con inputs y targets
- No hay un 'supervisor', o sea input y targets, hay una **señal de recompensa**
- El feedback **se recibe con retraso**, no es instantáneo
- Las decisiones son secuenciales, los datos no son i.i.d.
- Las acciones del agente **modifican** los datos que va recibiendo (el ambiente).

# EJEMPLOS

(EN CADA UNO DE ESTOS ¿RECOMPENSA, ACCIONES, OBSERVACIONES?)

- Un jugador de ajedrez, Teg, go, Backgammon, etc.
- `Un helicóptero debe realizar piruetas
- Diseñar landing page para maximizar retención
- Tratamiento médico personalizado
- Administración de una cartera de acciones
- Robots
- Asistentes de navegación

ALGUNAS DE LAS COMPLICACIONES



APRENDO DE LA EXPERIENCIA!

EXPLORACIÓN VS EXPLOTACIÓN

LOS POSIBLES ESTADOS DEL  
SISTEMA SON MUUUCHOS!

# VIDEOS DE ALGUNOS EJEMPLOS

- robot humanoide:

<https://www.youtube.com/watch?v=No-JwwPbSLA>

- helicoptero: <https://www.youtube.com/watch?v=0JL04JJjocc>

- blackout: <https://www.youtube.com/watch?v=eG1Ed8PTJ18>

- space invaders:

<https://www.youtube.com/watch?v=W2CAghUiofY>

- arquero robotico:

<https://www.youtube.com/watch?v=CIF2SBVY-J0>



# ÉXITOS

- TD-Gammon (1992)
- Atari Games (DQN, 2015)
- AlphaGo(2015/2016)/AlphaGo Zero(2017)/AlphaZero(2017)
- Dota 2 (2018)
- Starcraft 2 (2019)
- Manipulación Robótica (2018)

(<https://ai.googleblog.com/2018/06/scalable-deep-reinforcement-learning.html>)

El campo no ha hecho un impacto económico significativo aún, pero está comenzando a ser usado en diferentes industrias (grandes oportunidades). Tal vez el problema más importante: necesita ingentes cantidades de datos. Leer

<https://www.oreilly.com/ideas/practical-applications-of-reinforcement-learning-in-industry>

# COMPAÑÍAS UTILIZANDO APRENDIZAJE REFORZADO

- Deepmind: AlphaGo, AlphaZero, Atari Games, <https://deepmind.com/>
- Trading algorítmico: Hihedge, <https://www.hihedge.com/>, <https://pit.ai/>
- Ambientes de cultivo controlables: Optimal Labs: <http://optimal.ag/>
- Aprendizaje de robots/vehículos autónomos: <http://covariant.ai/>, <https://www.latentlogic.com/>, <https://www.osaro.com/>, <http://prowler.io/>, <https://www.fanuc.com/>
- Análisis de datos: <http://intelligentlayer.com/>
- Chatbots: <https://rasa.com/>

# REPASO DE PROBABILIDAD Y PROCESOS DE MARKOV

$\mathcal{L} = \phi \frac{L}{t} \frac{t}{\phi}$

$f(\omega) = \int_{-\infty}^{\infty} f(x) e^{-2\pi i x \omega} dx \quad \frac{d}{dt} \frac{d}{d\phi}$

$\nabla \cdot \mathbf{E} = 0 \quad \nabla \times \mathbf{E} = -\frac{1}{c} \frac{\partial \mathbf{H}}{\partial t} \quad \nabla \cdot \mathbf{H} = 0 \quad \nabla \times \mathbf{H} = \frac{1}{c} \frac{\partial \mathbf{E}}{\partial t}$

$(-i\hbar \frac{\partial}{\partial t} \Psi = H \Psi)$

$\rho \left( \frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v} \right) = -\nabla p + \nabla \cdot \mathbf{T} + \mathbf{f}$

$H = -\sum p(x) \log p(x)$

$\frac{1}{2} G^2 S^2 \frac{\partial^2 V}{\partial S^2} + r S \frac{\partial V}{\partial S} + \frac{\partial V}{\partial t} - r \cdot V = 0$

$(Q, q_i, m_i) = \sum_{i=1}^n \left[ \frac{D_i}{m_i q_i} S_i + c_i \cdot D_i + \frac{q_i H_i}{2} \left( m_i \left( 1 - \frac{D_i}{P_i} \right) - 1 + 2 \right) \right]$

$\left[ \frac{d \Delta p(s, \phi)}{d \phi} \right] = \begin{bmatrix} \gamma & -\mathcal{L} \\ -\beta & 0 \end{bmatrix} \begin{bmatrix} \Delta p(s, \phi) \\ \Delta M(s, \phi) \end{bmatrix}$

$\int_0^{\frac{\pi}{2}} (\log \sin x)^2 dx = \int_0^{\frac{\pi}{2}} (\log \cos x)^2 dx = \frac{\pi}{2} \left\{ \frac{\pi^2}{12} + (\log 2)^2 \right\}$

# BAYES

$$P(A|B) = P(B|A) \frac{P(A)}{P(B)}$$

Ejemplo Tiro un dado y después tiro una moneda tantas veces como el resultado del dado.

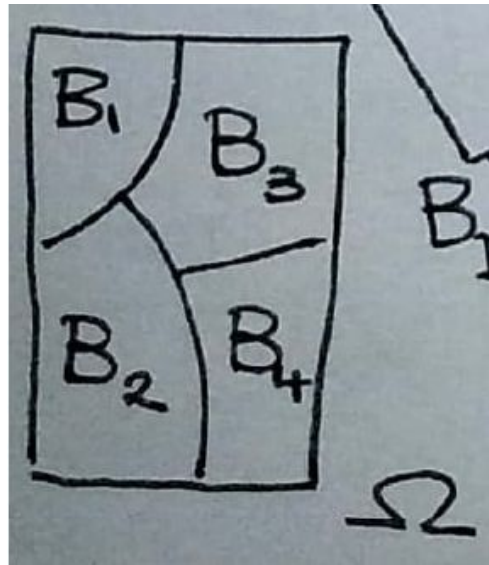
$D = \#$  dado ;  $M = \#$  de caras

Obs:  $M|D=d \sim \text{Bin}(d, \frac{1}{2})$

$$\begin{aligned} P(D=5|M=3) &= \frac{P(M=3|D=5) \cdot P(D=5)}{P(M=3)} \\ &= \frac{\binom{5}{3} \left(\frac{1}{2}\right)^3 \left(\frac{1}{2}\right)^{5-3} \times \frac{1}{6}}{P(M=3)} \end{aligned}$$

# FÓRMULA DE PROBABILIDAD TOTAL

$$\Omega = \bigcup B_k$$



$$P(A) = \sum_k P(A|B_k) \cdot P(B_k)$$

"posibles contextos"

$$\begin{aligned} P(M=3) &= \sum_{d=1}^6 P(M=3|D=d) P(D=d) \\ &= \sum_{d=3}^6 \binom{d}{3} \left(\frac{1}{2}\right)^d \times \frac{1}{6} \end{aligned}$$



# ESPERANZA CONDICIONAL

$$E[X] = \sum_x x P(X=x)$$

$P_B(A) := P(A|B)$  es una probabilidad

$$P_{X|Y=y}^{(x)} := \frac{P(X=x, Y=y)}{P(Y=y)} = \frac{P_{XY}(x,y)}{P_Y(y)} \quad \text{y está fijo!}$$

$X|Y=y \sim P_{X|Y=y} \quad \therefore \text{TIENE UNA ESPERANZA!}$

$$E[X|Y=y] = \sum_x x P_{X|Y=y}^{(x)}$$

VOLVIENDO AL EJEMPLO

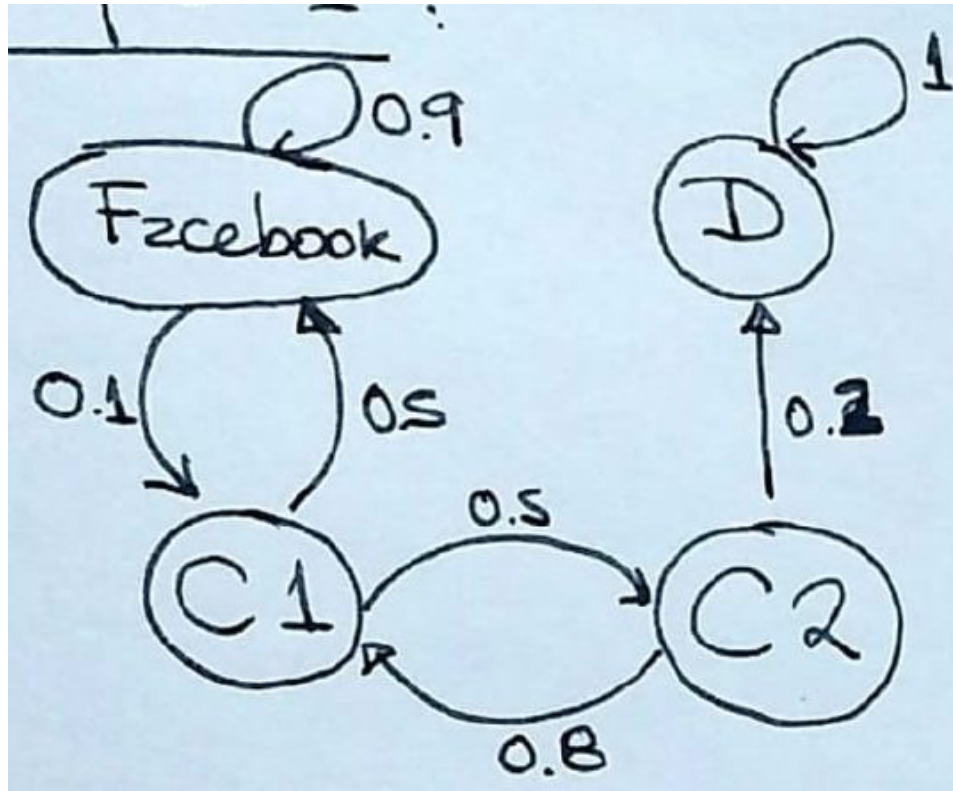
$$E[M | D=3] = \sum_{m=0}^3 m P_{M|D=3}^{(m)} = \sum_{m=0}^3 \binom{3}{m} \left(\frac{1}{2}\right)^3 m$$

## ESPERANZA DE LA ESPERANZA

$$\begin{aligned} E[X] &= E[E[X|Y]] \\ &= \sum_y E[X|Y=y] P_Y(y) \end{aligned}$$



# EJEMPLO DE CADENA DE MARKOV



	F	C <sub>1</sub>	C <sub>2</sub>	D
F	0.9	0.1	0	0
C <sub>1</sub>	0.5	0	0.5	0
C <sub>2</sub>	0	0.8	0	0.2
D	0	0	0	1

# OTRO EJEMPLO: REPAIR SHOP

SHOP

$n$  = # machines in the shop  
on day  $n$

$$X_{n+1} = \underbrace{(X_n - 1)^+}_{\substack{\text{c/ día 1} \\ \text{se repa}}} + \underbrace{Z_{n+1}}_{\text{entran 2 ser reparados}}$$

$$P(X_{n+1} = j | X_n = i) = P(j = (i-1)^+ + Z_{n+1})$$

entran 2 ser reparados

$$P(Z = k) = a_k ; \quad k \geq 0$$

$$P = \begin{pmatrix} a_0 & a_1 & \dots \\ a_0 & a_1 & \dots \\ 0 & a_0 & \dots \\ 0 & 0 & a_0 \dots \end{pmatrix}$$

## CONSTRUCCIÓN PARA ESPACIO DE ESTADOS FINITO

$$X_{n+1} = F(X_n, U_{n+1}), \quad U_{n+1} \perp\!\!\!\perp X_n$$

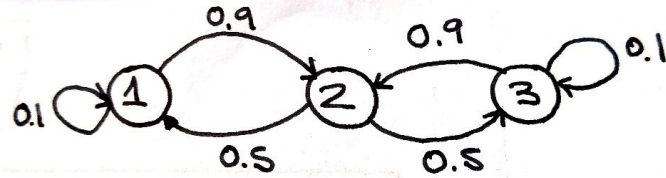
# DISTRIBUCIÓN INVARIANTE DE UNA CADENA DE MARKOV

$$P_{\mu}(X_1 = x) = \mu(x) \quad \forall x \in S$$

$$\text{ie; } \sum_y \mu(y) P(X_1 = x | X_0 = y) = \mu(x)$$

$$\boxed{\sum_y \mu(y) P(y, x) = \mu(x)}$$

$$\boxed{\mu P = \mu}$$



$$\mu(3) = \mu(1)$$

$$\mu(2) < \mu(1)$$

pues recibe mucho!  
(2)

# IRREDUCIBILIDAD

Communication:

•  $i \rightarrow j$  ( $j$  accessible from  $i$ ) if  $\exists k / p_{ij}^k > 0$ .

•  $i \leftrightarrow j$  (both are accessible from both)

DEF:  $P$  is irreducible if  $i \leftrightarrow j \forall i, j \in S$

# DEFINICIÓN DE RECURRENTE / TRANSIENTE

$$T_i = \inf \{ n \geq 1 ; X_n = i \}$$

$$i \in E \quad \text{recurrent} \quad P_i(T_i < \infty) = 1$$

$$\text{positive recurrent} \quad E_i[T_i] < \infty$$

$$\text{transient} \quad P_i(T_i < \infty) < 1$$

# TEOREMA ERGÓDICO PARA CADENAS DE MARKOV (BRÉMAUD)

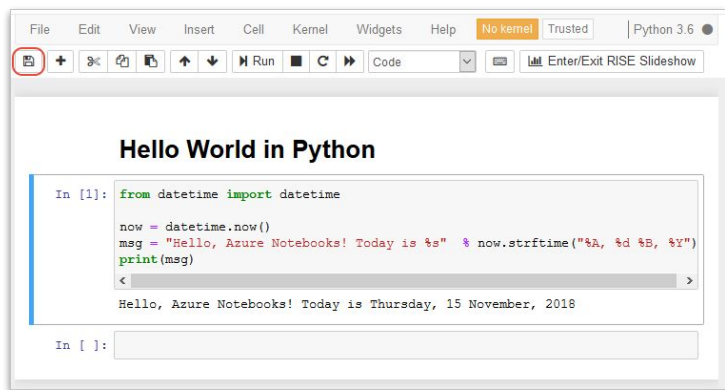
## THM 4.1 ERGODIC THM

$\{X_n\}_n$  irreducible, positive recurrent MC

$\forall \gamma$  initial dist,  $P_\gamma$ -as

$$\Rightarrow \lim_{k \rightarrow \infty} \frac{\sum_{n=1}^k f(X_n)}{k} = E_\mu[f(X)]$$
$$\sum_i \mu(i) f(i)$$

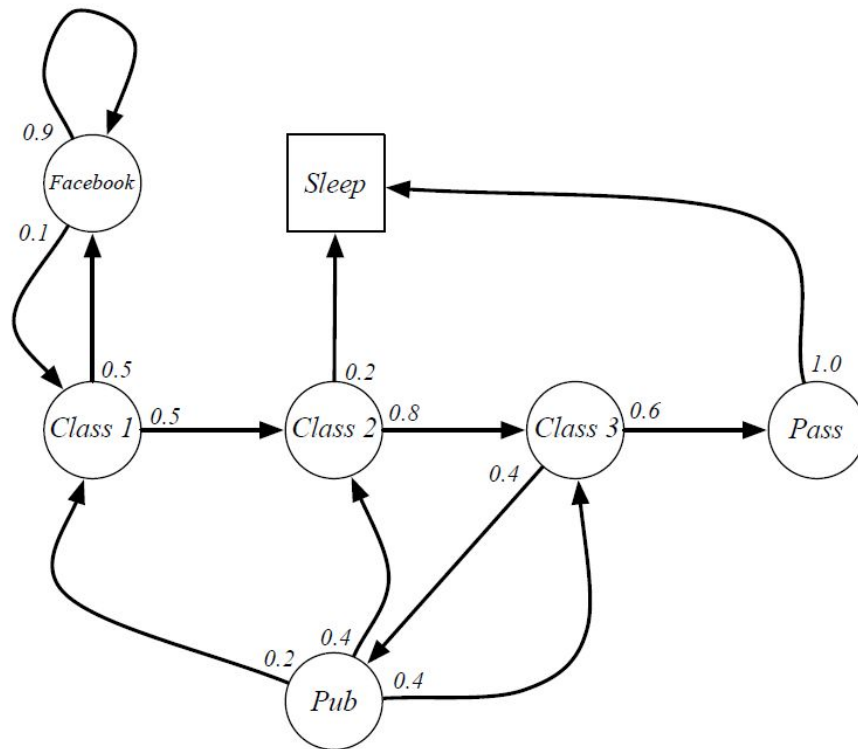
# EJEMPLO DEL ESTUDIANTE



The screenshot shows the Azure Notebook interface with a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar. A red circle highlights the 'Run' button (a play icon). Below the toolbar, the title 'Hello World in Python' is displayed. The code cell contains the following Python code:

```
In [1]: from datetime import datetime
now = datetime.now()
msg = "Hello, Azure Notebooks! Today is %s" % now.strftime("%A, %d %B, %Y")
print(msg)
<
Hello, Azure Notebooks! Today is Thursday, 15 November, 2018
```

The output of the code is displayed below the code cell.



Tomado de los slides de David Silver

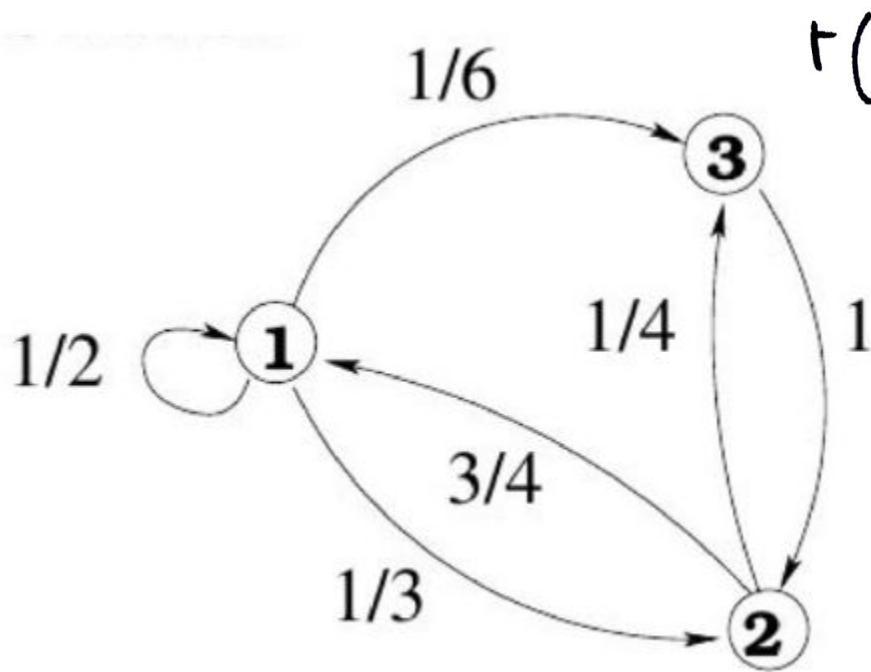


# EJERCICIO 1.1 - ESTUDIANTE - PYTHON

**Usando el código hacer los siguientes ejercicios:**

1. simular 100 episodios del estudiante
2. calcular un estimado del tiempo de visita de cada estado
3. calcular un estimado de la longitud media de la cadena

## EJERCICIO 1.2 - MC SIMPLE CON REWARD - PAPEL Y PYTHON



$$r(1) = -2, \quad r(2) = 3, \quad r(3) = 5$$

$$g(s_1, s_2) = r(s_1) + r(s_2)$$

Calcular esto de manera analítica y vía simulación:

$$E_1[g(s_1, s_2)]$$

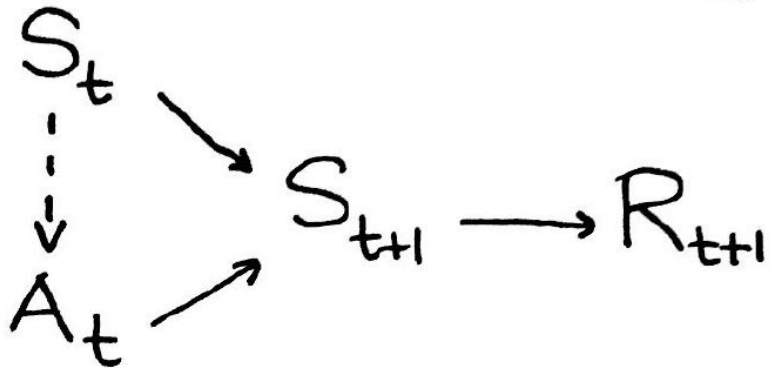
# PROCESOS MARKOVIANOS DE DECISIÓN

$\mathcal{L} = \phi \frac{\partial}{\partial t}$   
 $\nabla \cdot \mathbf{E} = 0 \quad \nabla \times \mathbf{E} = -\frac{1}{c} \frac{\partial \mathbf{H}}{\partial t}$   
 $\nabla \cdot \mathbf{H} = 0 \quad \nabla \times \mathbf{H} = \frac{1}{c} \frac{\partial \mathbf{E}}{\partial t}$   
 $-\hbar \frac{\partial}{\partial t} \Psi = H \Psi$   
 $f(w) = \int_{-\infty}^{\infty} f(x) e^{-2\pi i x w} dx \quad \frac{d}{dt}$   
 $\rho \left( \frac{\partial v}{\partial t} + v \cdot \nabla v \right) = -\nabla p + \nabla \cdot \mathbf{T} + \mathbf{f}$   
 $H = -\sum p(x) \log p(x)$   
 $\frac{1}{2} G^2 S^2 \frac{\partial^2 V}{\partial S^2} + r S \frac{\partial V}{\partial S} + \frac{\partial V}{\partial t} - r \cdot V = 0$   
 $(Q, q_i, m_i) = \sum_{i=1}^n \left[ \frac{D_i}{m_i q_i} S_i + c_i v D_i + \frac{q_i H_i}{2} \left( m_i \left( 1 - \frac{D_i}{P_i} \right) - 1 + 2 \right) \right]$   
 $\left[ \begin{array}{c} \frac{d \Delta p(s, \phi)}{d \phi} \\ \frac{d \Delta M(s, \phi)}{d \phi} \end{array} \right] = \left[ \begin{array}{cc} \gamma & -\mathcal{L} \\ -\beta & 0 \end{array} \right] \left[ \begin{array}{c} \Delta p(s, \phi) \\ \Delta M(s, \phi) \end{array} \right]$   
 $\int_0^{\frac{\pi}{2}} (\log \sin x)^2 dx = \int_0^{\frac{\pi}{2}} (\log \cos x)^2 dx = \frac{\pi}{2} \left\{ \frac{\pi^2}{12} + (\log 2)^2 \right\}$

# PROCESOS MARKOVIANOS DE DECISIÓN - INGREDIENTES

POLÍTICA  $\pi(a|s) = P(A_t = a | S_t = s)$

$$P_{s,s'}^a := P(S_{t+1} = s' | S_t = s, A_t = a)$$



¿LA POLÍTICA ÓPTIMA?

FACTOR DE DESCUENTO

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots$$

$$= R_{t+1} + \gamma G_{t+1}$$

RETORNO

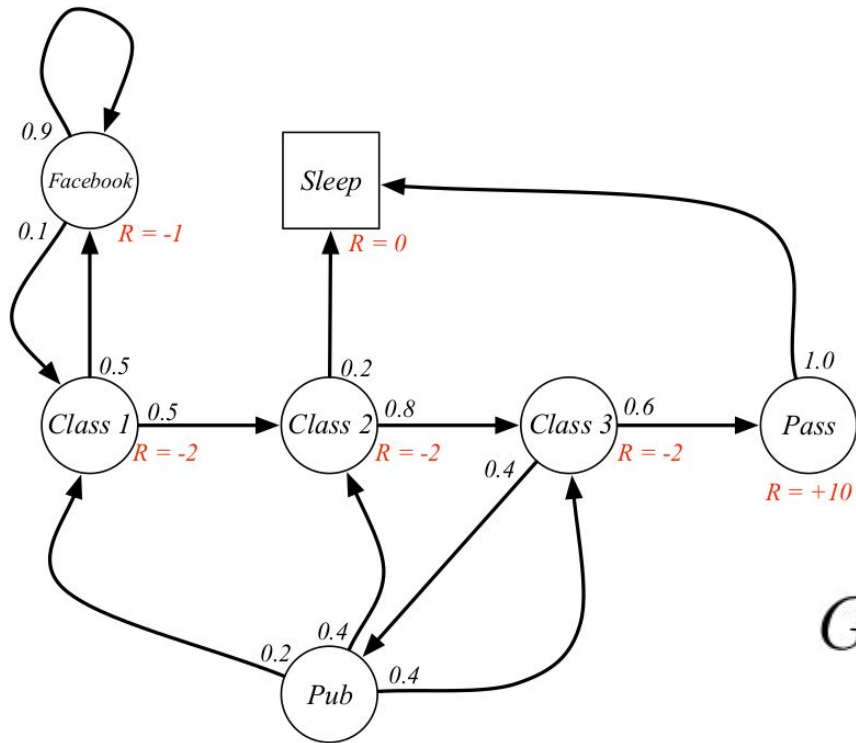
$$v_{\pi}(s) := E_{\pi}[G_t | S_t = s]$$

FUNCIÓN  
DE VALOR

# REWARD HYPOTHESIS

That all of what we mean by goals and purposes can be well thought of as the maximization of the expected value of the cumulative sum of a received scalar signal (called reward).

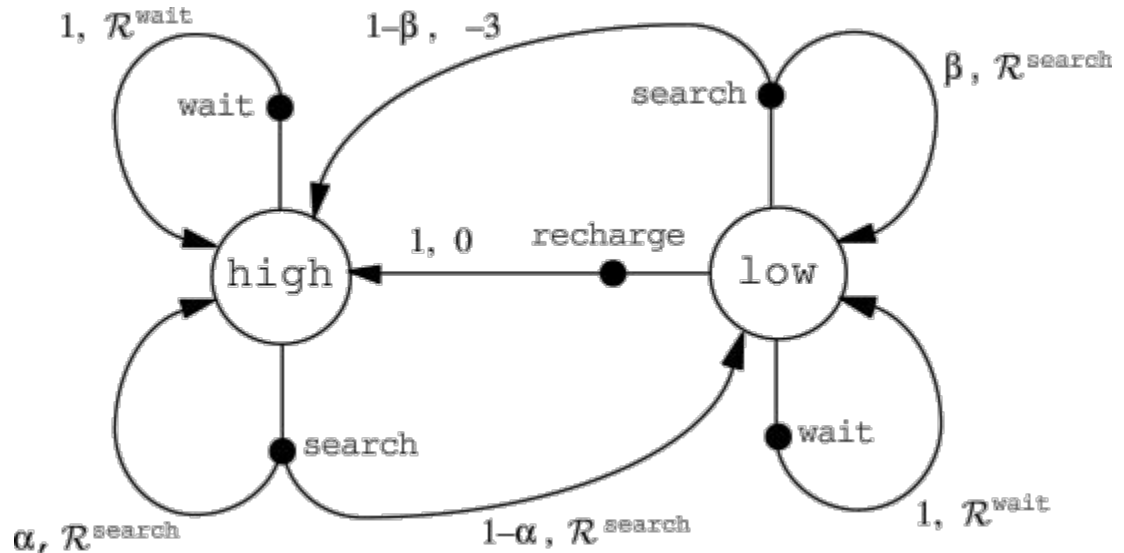
# EPISODIC TASK VS CONTINUING TASK



$$G_t \doteq \sum_{k=t+1}^T \gamma^{k-t-1} R_k,$$

# EJEMPLO DEL ROBOT

$$r(s,a,s') = \mathbb{E} [R_{t+1} \mid S_t = s; A_t = a; S_{t+1} = s']$$




Tomado de los slides del libro de Sutton



# EJEMPLO MÁS COMPLEJO - ALQUILER DE AUTOS

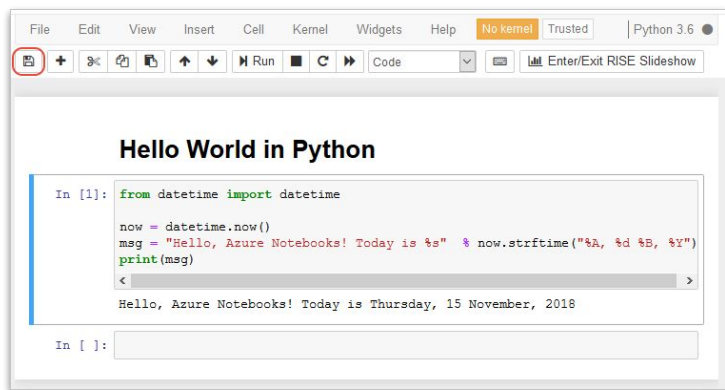
- Dos terminales, A y B.
- Un máximo de 20 autos por terminal.
- Puedo mover máximo en cada noche 5 autos de una terminal a otra. Cada auto cuesta 2\$ moverlo.
- La cantidad de autos *demandados* en cada una de las terminales sigue una distribución de Poisson de medias 3 y 4 respectivamente.
- La cantidad de autos *retornados* en cada una de las terminales sigue una distribución de Poisson de medias 2 y 3 respectivamente.
- Cada auto alquilado da una ganancia de 10\$.
- *Si alguna de las dos terminales se queda sin autos se acaba el negocio.*

## EJERCICIO 1.3 - RATA 5.14 - PAPEL

**5.13**  Una **rata** está atrapada en un laberinto. Inicialmente puede elegir una de tres sendas. Si elige la primera se perderá en el laberinto y luego de 12 minutos volverá a su posición inicial; si elige la segunda volverá a su posición inicial luego de 14 minutos; si elige la tercera saldrá del laberinto luego de 9 minutos. En cada intento, la rata elige con igual probabilidad cualquiera de las tres sendas. Calcular la esperanza del tiempo que demora en salir del laberinto.

**5.14** Una rata está atrapada en un laberinto. Inicialmente elige al azar una de tres sendas. Cada vez que vuelve a su posición inicial elige al azar entre las dos sendas que no eligió la vez anterior. Por la primera senda, retorna a la posición inicial en 8 horas, por la segunda retorna a la posición inicial en 13 horas, por la tercera sale del laberinto en 5 horas. Calcular la esperanza del tiempo que tardará en salir del laberinto.

# INTRODUCCIÓN A OPENAI GYM



- Introducción General
- Para que “jueguen”: Mountain
- Ejemplo del Robot (Batería)

# INTRODUCCIÓN A OPENAI GYM

- ¿Cómo instalarlo? ubuntu 18.04, Python, jupyter, open ai gym
  - Linux:
    - `sudo apt-get update`
    - `sudo apt-get install python3 python3-pip ipython3 python3-fontconfig`
    - `sudo apt-get install libglu1-mesa-dev freeglut3-dev mesa-common-dev python-opengl`
    - `pip3 install numpy pandas matplotlib jupyter gym`
  - Windows:
    - Virtual Box:
      - instalar ubuntu 16.04 y usar el instructivo de la parte de Linux
    - WSL:(inspirado en <https://github.com/openai/gym/issues/11#issuecomment-242950165>)
      - instalar Windows Subsystem for Linux (WSL): <https://docs.microsoft.com/en-us/windows/wsl/install-win10>
      - instalar ubuntu 16.04 LTS para WSL yendo a Microsoft Store (barra de búsqueda de Windows) y buscando Ubuntu 16.04
      - correr una consola WSL (buscar Ubuntu en la barra de búsqueda de Windows)
      - realizar los mismos pasos que en el instructivo de linux en esa consola
      - instalar vcXsrv/xming;
      - correr vcXsrv (elegir one large window); tipear en la consola de comandos de WSL: `export DISPLAY=:0`
  - Correr jupyter: `jupyter notebook --no-browser`
  - Google colab: ir a google colab, <https://colab.research.google.com/notebooks/welcome.ipynb#recent=true>, elegir la solapa Github y buscar en <https://github.com/javkrei/aprendizaje-reforzado-austral>

# LECTURAS RECOMENDADAS

- AlphaGo paper: <https://ai.google/research/pubs/pub44806>
- Brief Survey of Deep RL: <https://arxiv.org/pdf/1708.05866.pdf>
- Sutton capítulo 1 para una introducción, capítulo 16 para aplicaciones, 14 y 15 para relación con psicología y neurociencia.