Reinforcement Learning & HMM

Instituto Tecnológico Autónomo de México

Primavera 2017

Introducción

Contacto

Instructor: Mauricio Benjamín García Tec

Correo: mauricio.garcia@itam.mx

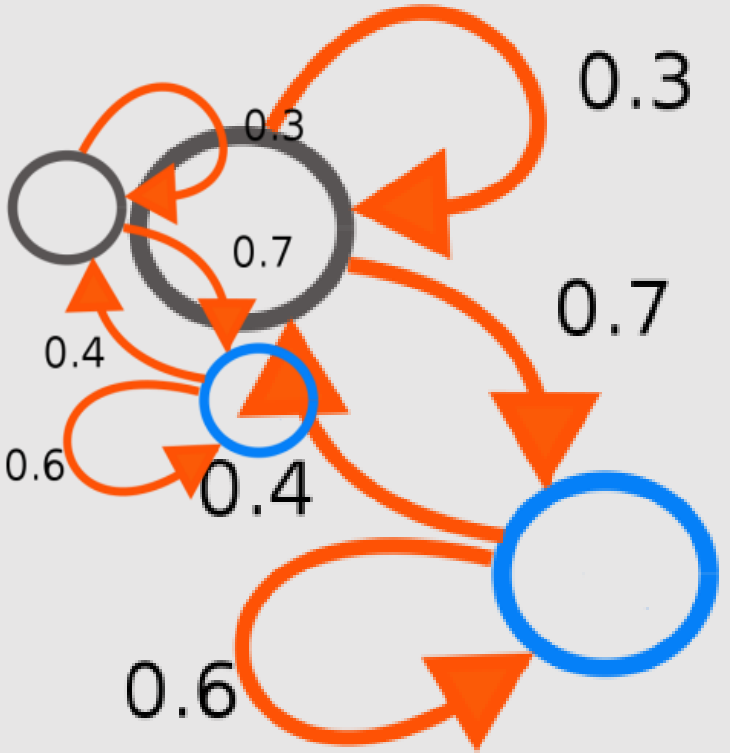
Oficina: Pasillo de actuaría, 1er piso, Río Hondo

Temario, página del curso y materiales:
https://github.com/mauriciogtec/reinforcementHMM2017.git

¿De qué es esta clase?

- Esta clase está divida en dos secciones: cada una es una aplicación avanzada de las Cadenas de Markov
  - I: Reinforcement Learning: ~70%
  - II: Hidden Markov Chain Models (HMM): ~30%

- Reinforcement Learning se trata de aprender a tomar decisiones a través de ensayo y error. Para eso se extienden los Procesos de Markov a Procesos Markovianos de Decisión
- Los HMMs se tratan de cadenas de Markov que no podemos ver, pero cuyos estados ocultos podemos adivinar observando otro grupo de estados visibles
- Los Procesos Markovianos de Decisión Parcialmente Observables son la unión de los dos mundos (platicaremos más sobre ellos pero no los veremos con detalle)



¿Poqué Markov?

- Un **PROCESO ESTOCÁSTICO** es **MARKOVIANO** si su estado actual determina los posibles estados futuros del proceso y la probabilidad de estar en ellos
- La mayoría de los **SISTEMAS** se pueden **MODELAR** markovianamente

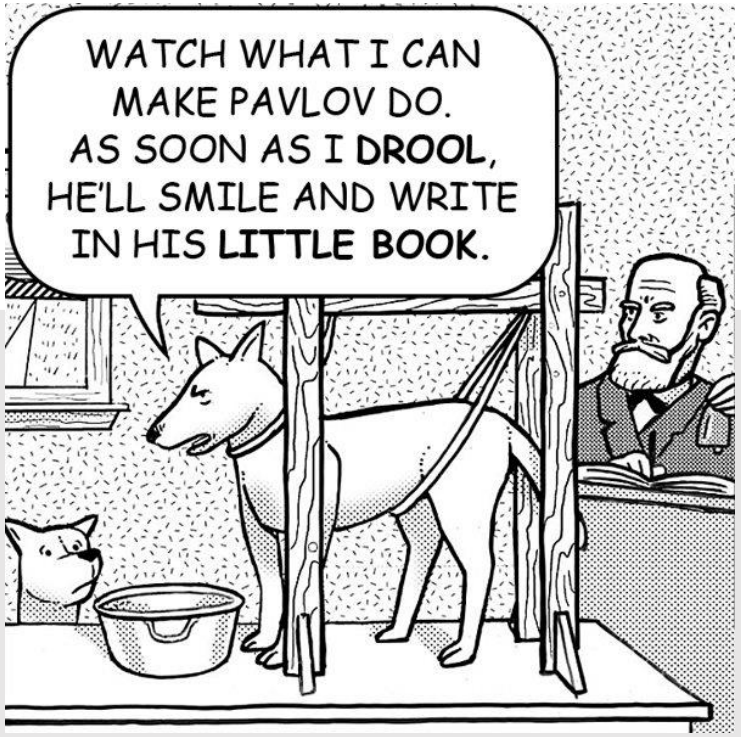

indexes - day, 5
indexes - day, 12
61.81%
50.00%
38.19%
23.88%

Inteligencia Artificial

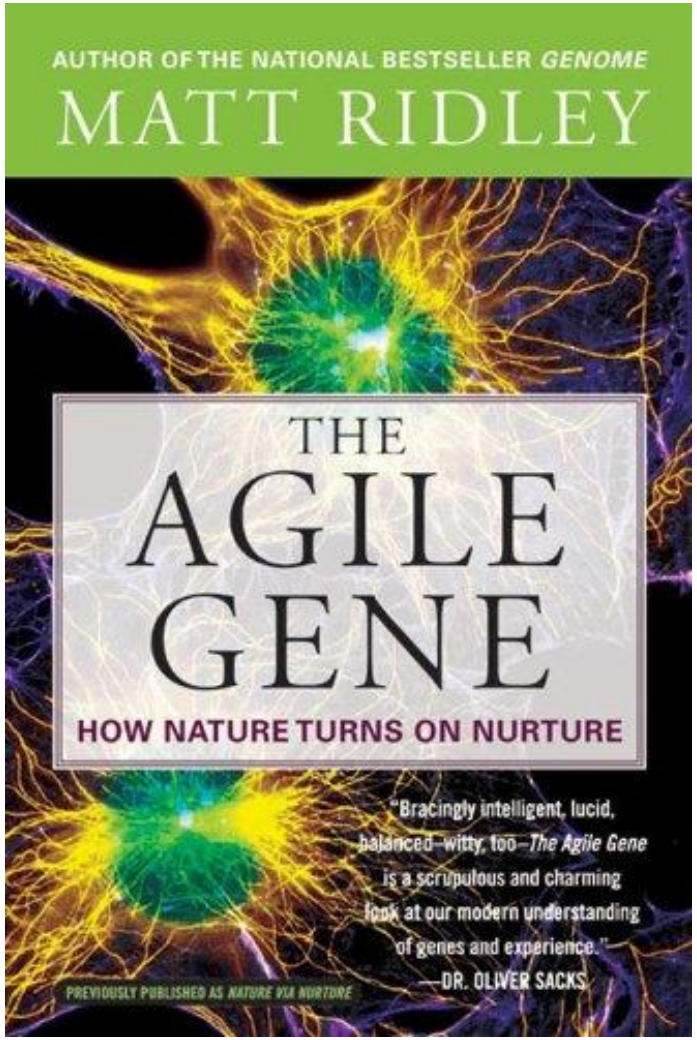
- También la forma en que aprendemos se puede modelar como una cadena de Markov…
- Muchas de las aplicaciones del curso serán a Inteligencia Artificial
- Esto nos da pie a entrar de lleno al mundo del Reinforcement Learning….

Reinforcement Learning

- ¿Qué es el reforzamiento?
- ¿Cómo aprendemos?



- La mayoría de lo que vamos a prender viene de nuestro entendimiento de cómo aprenden los animales y los seres humanos



I. El problema del Reinforcement Learning

Los elementos de un problema de RL

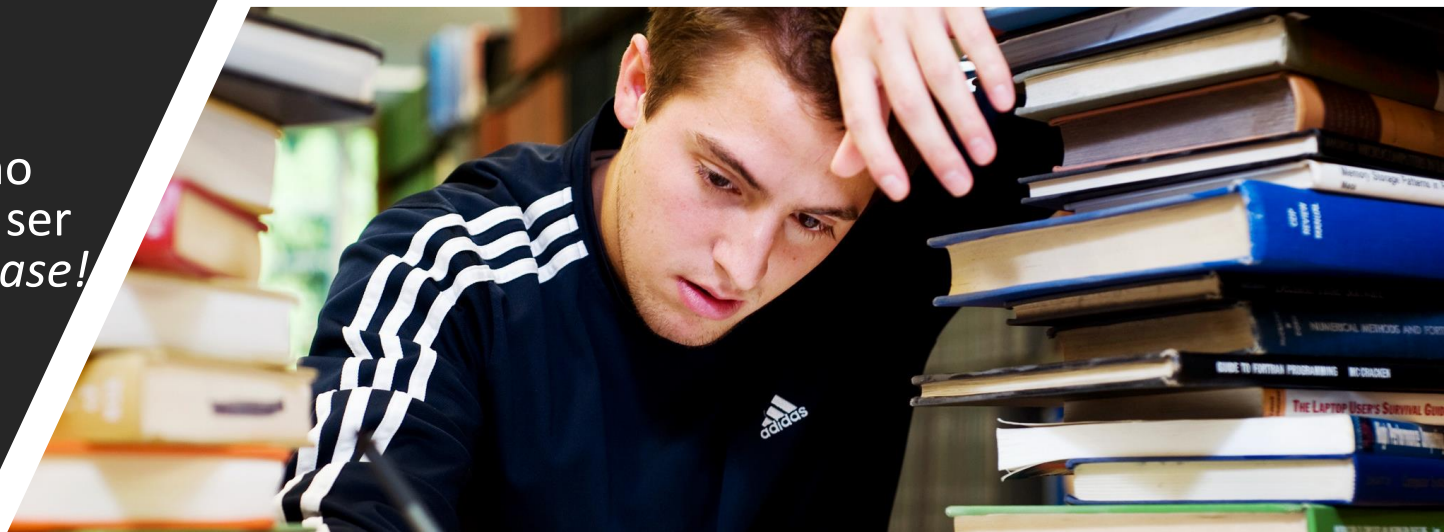
1. Una política de acciones (*policy*): reglas de toma de decisión
2. Premio (*reward signal):* es el castigo o beneficio, medido numéricamente por tomar UNA decisión
3. Una función de valor (*value function)*: es todo el premio acumulado que se puede derivar de tomar una acción
4. [optativo] Un modelo el ambiente (*environment model)*: reprenta un cambio en las condiciones de aprendizaje

Reward vs Value

- El *reward* o premio, es el beneficio **INMEDIATO** de tomar una acción

- El *value* de una acción incluye el beneficio inmediato **Y** todo lo que se pueda acumular en futuro de esa decisión.

- Por ejemplo, algunas acciones que hoy no son muy divertidas, en el futuro pueden ser de gran valor, *como estudiar para esta clase!*



- Algunas acciones claro no tienen ni *reward* ni *value…*
- Pero esta es una clase donde que se inspira del *rational-decision-making,* **así que esto no va a pasar!!!!!!!!!!!!!!!!!**

Modelo del ambiente = **PLANEACIÓN**

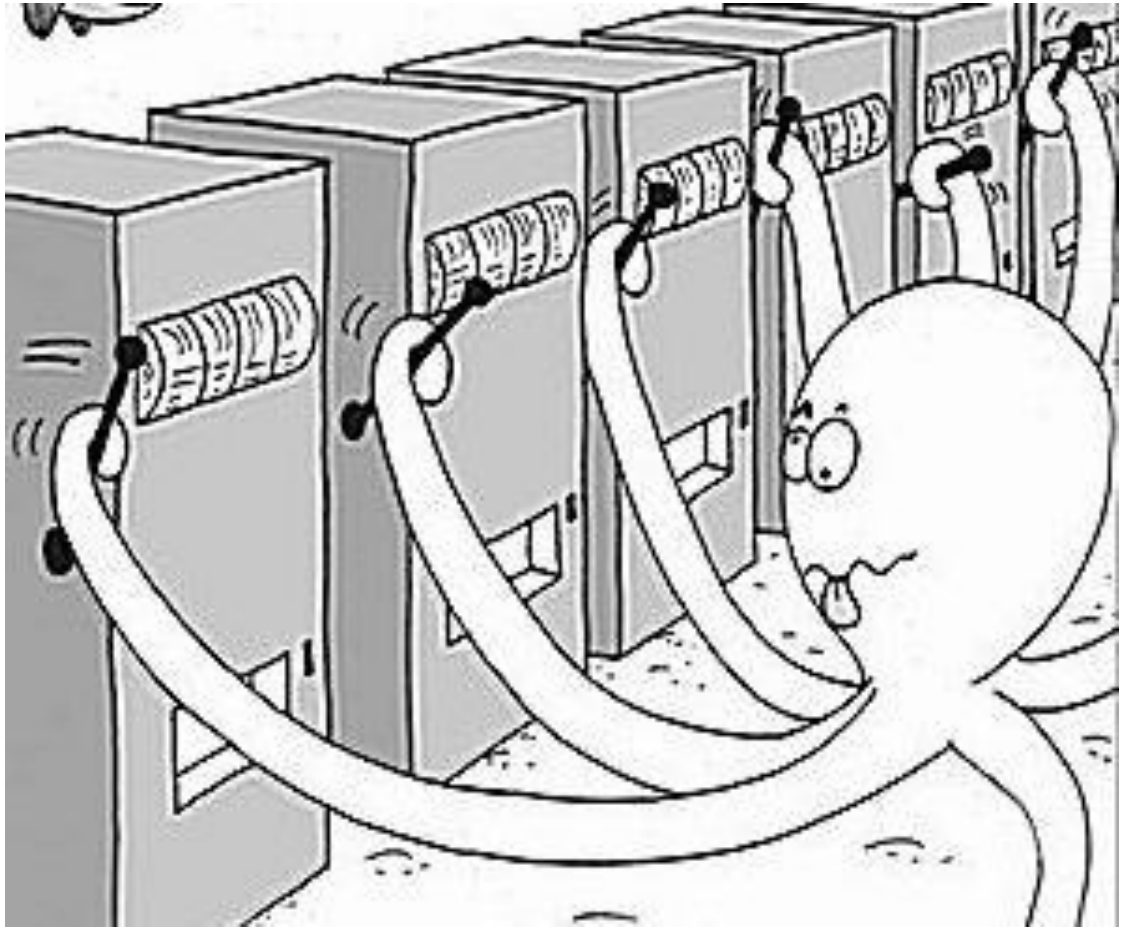
- En algunos problemas, las condiciones son cambiantes.
- Por ejemplo, decidir cuánto ahorrar: depende de si Trump es presidente o no….
- El modelo el ambiente incluye esto: un modelo predictivo de las situaciones que influyen el valor de las decisiones
- O llenar el tanque si va a haber gasolinazo
- *PLANNING*

RL vs Otro tipos de aprendizaje

- RL ≠ Supervised Learning
- RL ≠ Unsepervised Learning

- Es verdaderamente otro tipo de aprendizaje
- La diferencia es que **NO** hay una colección de ejemplos del cual aprender
- Aprendemos de **evaluar** el resultado de tomar una acción
- No hay un experto del cual aprender
- Nosotros no somos el **experto,** debemos enseñar a aprender

II. Introducción a los métodos de solución: *The multi-armed bandit*



El *multi-armed bandit*

- El *multi-armed* bandit o el problema de las *k*-cubetas busca maximizar el ingreso al elegir entre un conjunto de *k* opciones a lo largo del tiempo

- El problema del *multiarmed bandit* es interesante porque evidencia el primer reto en el curso: **balancear exploración vs explotación**

Estrategia 1: Algoritmos *Greedy* y ϵ-Greedy

- Un algoritmo **greedy** o codicioso es aquel en el que siempre se elije la opción que ha promedio el mayor premio.
- Un algoritmo **ϵ-greedy** escoje la opción **codiciosa** una proporción $(1-\epsilon)\%$ de las veces y el resto escoje aleatoriamente entre todas las opciones.
  - Variantes de este algoritmo cambian la forma de exploración (e.g., en vez de explorar todas las opciones, puede usarse la segunda major)