Parte01

October 21, 2018

1 Introducción al Aprendizaje por Refuerzos

1.0.1 Curso Aprendizaje por Refuerzos, Diplomatura en Ciencia de Datos, Aprendizaje Automático y sus Aplicaciones

1.0.2 FaMAF, 2018

Agenda Clase 1 Parte 1

- Presentación Docentes
- Introducción. Modelo Agente Entorno. Agente Situado. Arquitectura Actor-Crítico.
- Aprendizaje por Refuerzos. Elementos. Ciclo del Aprendizaje por Refuerzos. Definición Formal.
- Procesos de Decisión de Markov. Función de Valor. Ecuación de Bellman. Optimalidad.
- Aproximaciones al Aprendizaje. Model Free y Model Based.
  - Iteración de Política.
  - Iteración de Valor.

1.1 Docentes

- Jorge A. Palombarini
- Juan Cruz Barsce
- Ezequiel Beccaría

1.2 Página y Libro Sutton

http://incompleteideas.net/
https://drive.google.com/file/d/1opPSz5AZ_kVa1uWOdOiveNiBFiEOHjkG/view

1.3 Link Slack

https://diplodatos2018.slack.com/messages/CD8LWUL3C/

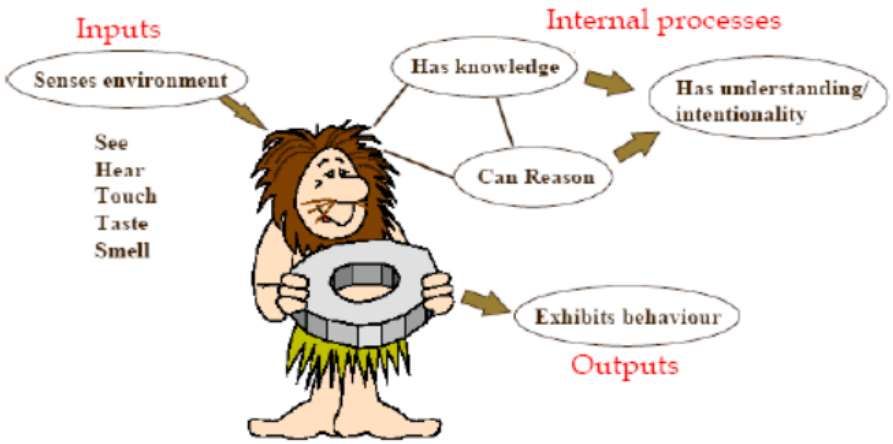
1.4 Introducción: Entidad Inteligente -> Agente Situado

- La idea de aprender por interacción con nuestro entorno es quizás la primera en aparecer cuando pensamos acerca de la naturaleza del aprendizaje.

- Por ejemplo, cuando un niño juega, mueve sus brazos, o mira alrededor, no tiene un "maestro" explicito, pero posee una conexion sensorial y motora con su entorno.
- El ejercicio de dicha conexión produce información acerca de la relación causa-efecto y las consecuencias de las acciones que lleva a cabo, y respecto de qué hacer de manera tal de lograr objetivos.
  Así, *aprender por interacción* es una idea fundacional de muchas teorías del aprendizaje y la inteligencia.
- En este curso, se explorará un enfoque computacional dirigido por objetivos para *aprender por interacción*: el **Aprendizaje por Refuerzos (Reinforcement Learning)**.

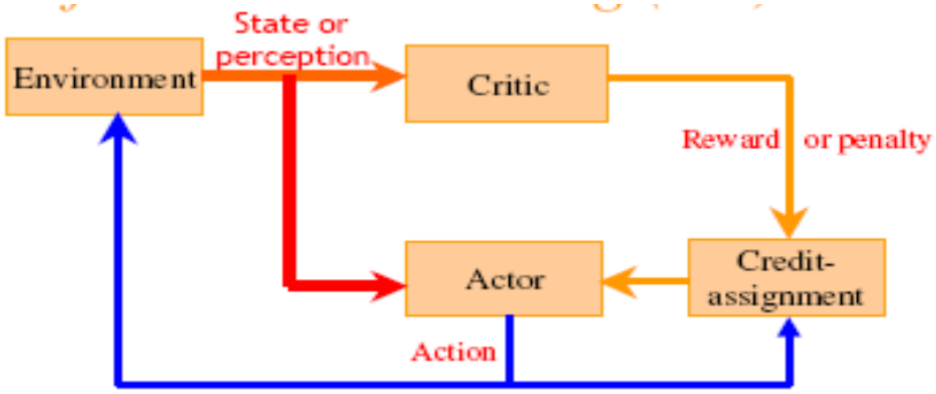
1.5 Agent-Environment Framework

- El desarrollo de la inteligencia requiere que la entidad o el agente esté situada/o en un entorno **(Measuring universal intelligence: Towards an anytime intelligence test, Hernandez-Orallo & Dowe, Artificial Intelligence, 2010).**



- El agente y su entorno interactúan a través de la ejecución de acciones, observación de estados y señales de reward.
  La inteligencia tendrá efecto sólo si el agente tiene claramente definidos objetivos o metas que persigue activamente mientras ocurre la interacción.

1.6 Arquitectura Actor-Crítico

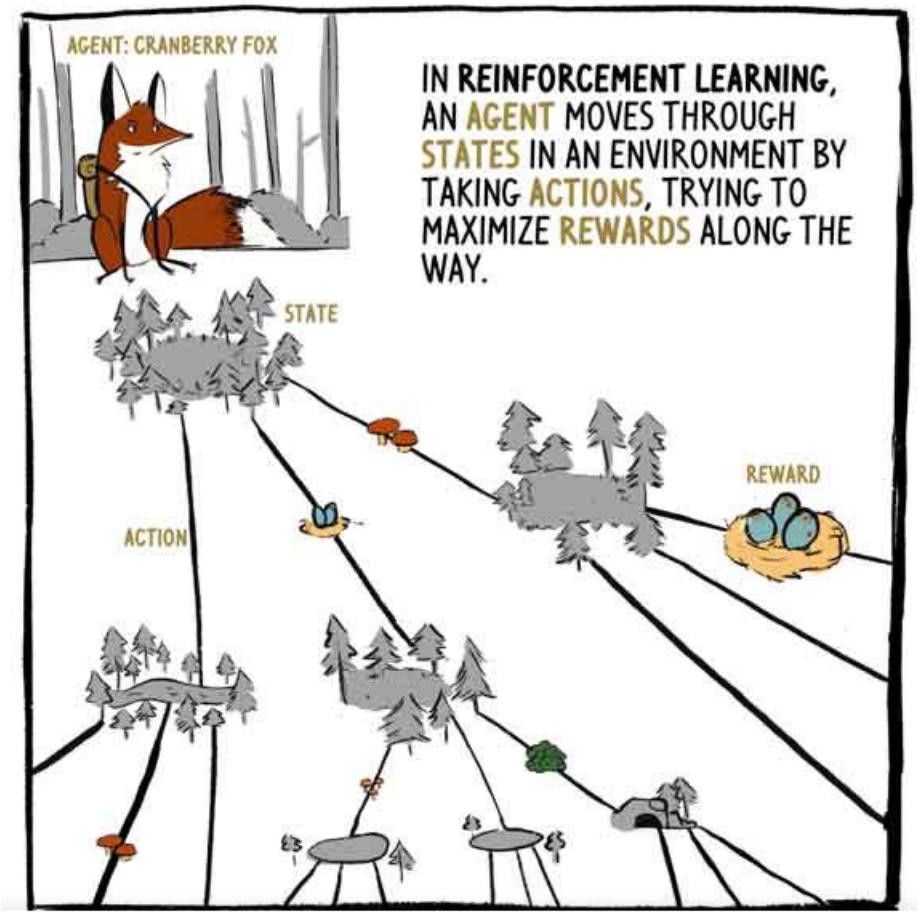


1.7 Reinforcement Learning

- ***Reinforcement learning*** consiste en aprender **que hacer** -mapear situaciones a acciones- de manera tal de **maximizar** una señal numérica de recompensa.
- Al aprendiz no se le especifica cuáles acciones ejecutar, sino que debe descubrir a través de ***prueba y error*** cuáles acciones producen la mayor cantidad de recompensa acumulada.
  En los casos más interesantes, las acciones pueden afectar no solo la recompensa inmediata, sino solamente el estado, recibiéndose una recompensa sólo en algunos estados o bien al final del episodio (***Delayed-reward***).
- Se conoce simultaneamente como RL al problema de aprender por interacción, a la clase de métodos de solución de dicho problema, y al área de la IA que estudia el problema y los métodos de solución.
- El problema de RL puede ser formalizado empleando **Procesos de Decisión de Markov**.
- La toma de decisiones secuencial involucra aprender sobre nuestro entorno y elegir acciones que maximizan el retorno esperado.
  El RL computacional, inspirado por estas ideas, las formalizo y produjo un impacto importante en robótica, machine learning y neurociencias.
- El Aprendizaje por Refuerzos (RL) consiste en un agente que se encuentra en algún estado $s \in S$ inmerso en un entorno E y toma acciones $a \in A$ en busca de una meta.
  El agente puede ser modelado formalmente como una función f, que toma un historial de interacción como entrada, y devuelve una acción a tomar.
  Una manera conveniente para representar el agente es una medida de probabilidad sobre el set A de acciones, en base a un historial de interacción:

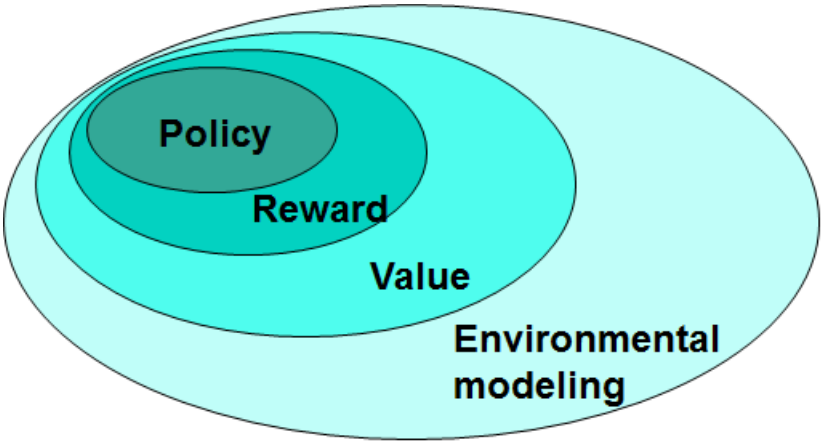
  $$f(a_n|s_1a_1r_2s_2a_2...r_ns_n)$$

  que representa la probabilidad de la acción a en el ciclo n dado un historial de interacción.



- Problema RL: ***£Cómo el agente produce la distribución de probabilidad sobre las acciones?***
- ***Dilema de exploración - explotación***: debido a que el Agente no recibe ejemplos de entrenamiento, debe probar alternativas, procesar los resultados de sus acciones y modificar su comportamiento en algún sentido. £Cuándo explotar este conocimiento vs. cuándo probar nuevas estrategias?

1.8 Elementos del Aprendizaje por Refuerzos



- **Policy (Política):**

Una política define la manera de comportarse de un agente, en cualquier momento de tiempo dado.
Basicamente, es un mapeo de un estado o percepción s a una acción a, pudiendo ser estocásticas.

- **Reward Function (Función de Recompensa)**

Define cuantitativamente el objetivo del agente.
Es un mapeo de un par estado-acción a un número real que indica "cuán deseable" es ejecutar dicha acción en ese estado.
Asimismo, el único objetivo del agente es maximizar la recompensa total que recibe a lo largo del tiempo.
Cabe mencionar que, si bien la función de reward no puede ser alterada por el agente, provee las bases para cambiar la política del mismo.

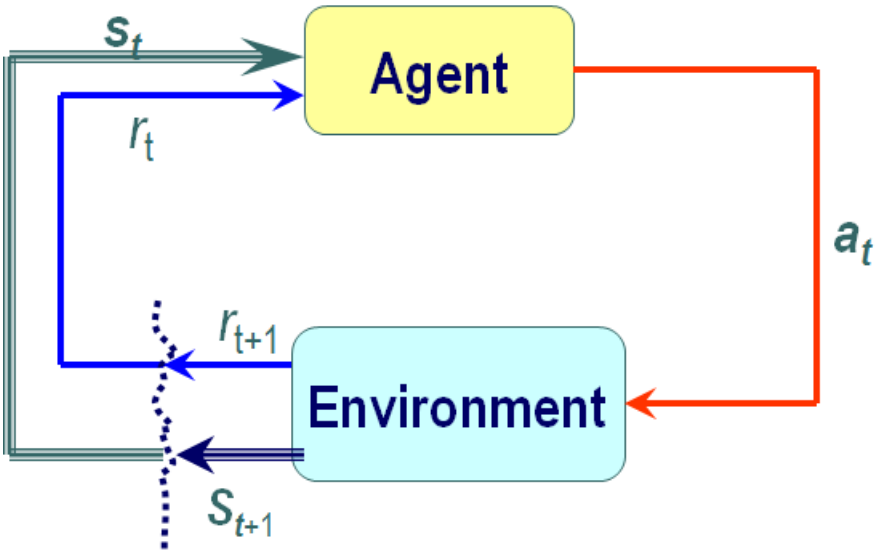
- **Value function (Función de Valor)**

La función de valor se diferencia de la función de reward en el sentido de que indica "cuán deseable" es, a largo plazo, ejecutar una acción en un determinado estado.
Así, el valor de un estado s es la cantidad total de reward que el agente espera obtener a futuro comenzando la interacción en el estado s.

- **Environment (Entorno)**

El entorno se encuentra constitutido por todo aquel elemento (real o simulado) que el agente no puede controlar.
Es con quién el agente interactúa a partir de la ejecución de acciones de control.

1.9 Ciclo del Aprendizaje por Refuerzos



1.9.1 Definición formal

- Si el problema de RL dado tiene un conjunto finito de estados y acciones y satisface la propiedad de Markov entonces puede definirse como un Proceso de Decisión de Markov

$$MDPFinito = (S, A, P(.), R(.),) \tag{1}$$

donde

$$S = s_1, s_2, ..., s_n$$

es un conjunto finito de estados.

$$A = a_1, a_2, ..., a_m$$

es un conjunto finito de acciones.

$$P_a(s, s') = P(s_{t+1} = s | s_t = s, a_t = a)$$

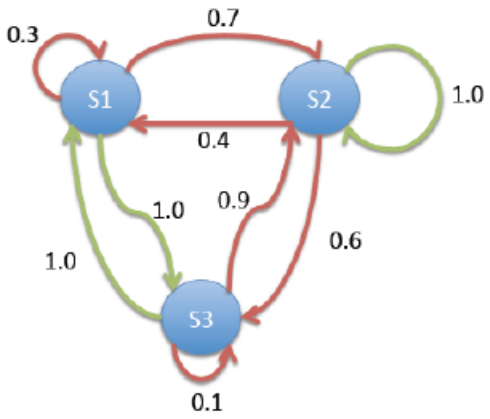
es la probabilidad de que la acción a tomada en tiempo t y en estado s lleve al agente al estado s' en tiempo t+1

$$R_a(s, s')$$

es la recompensa inmediata recibido tras transicionar, luego de tomar la acción a, desde el estado s al estado s'

$$\gamma \in [0, 1]$$

es el factor de descuento, representando la diferencia en la importancia de la recompensa a corto plazo vs la recompensa a largo plazo.



R(s1) = +1
R(s2) = 0
R(s3) = -1

Función de transición T:

S	A	S'	p
S1	R	S1	0.3
	R	S2	0.7
	V	S3	1.0
S2	R	S1	0.4
	R	S2	0.6
	V	S2	1.0
S3	V	S1	1.0
	R	S3	0.1
	R	S2	0.9

- Un episodio (instancia) de este MDP forma una secuencia finita

  $$s_0, a_0, r_1, s_1, a_1, r_2, s_2, ..., s_{n-1}, a_{n-1}, r_n, s_n$$

  donde

  $$s_n$$

  es un estado final (o n es el tiempo de corte).

- La recompensa total del episodio está dado por

  $$R = r_1 + r_2 + ... + r_n$$

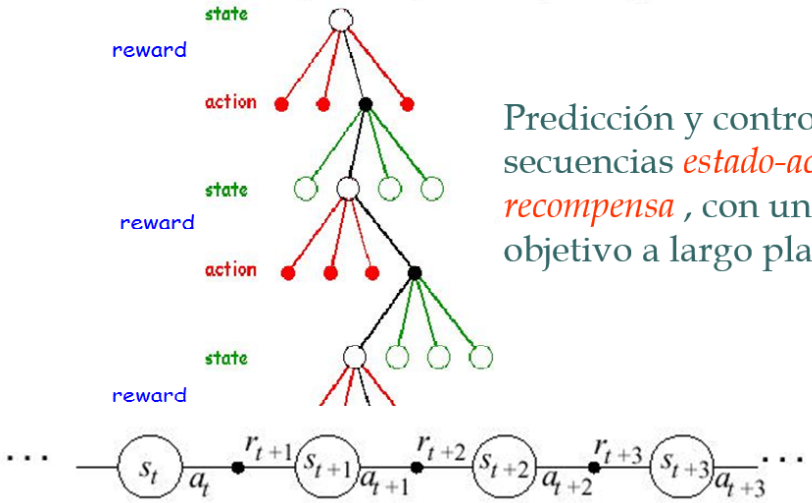
- En consecuencia, la recompensa a futuro partiendo del tiempo t está dado por

  $$R_t = r_t + r_{t+1} + ...$$

- Hay que considerar que el ambiente es estocástico en la mayor parte de los entornos reales y, por tanto, la recompensa suele diverger mientras más alejado se encuentre el instante de tiempo considerado. Es por esto que se utiliza un parámetro llamado *factor de descuento*, para descontar el valor de las recompensas futuras. De esta manera,

$$R_t = r_t + r_{t+1} + {}^2r_{t+2} + {}^3r_{t+3} + ... = r_t + (r_{t+1} + r_{t+2} + {}^2r_{t+3}...) = r_t + R_{t+1} \quad (2)$$

- Si utilizamos $\gamma = 0$, el agente priorizará sólo la recompensa inmediata, mientras que $\gamma = 1$ hará que considere todas los recompensas de la misma manera, independientemente del momento en donde las reciba.



- Dado un Proceso de Decisión de Markov, una Política (determiniística) es una función π que a partir de un estado $s \in S$, devuelve como resultado una acción $a \in A$. Una política estocástica devuelve, para un estado s y una acción a, una probabilidad.

1.10 Procesos de Decisión de Markov

1.10.1 Función de Valor

- El valor de un estado es el retorno esperado por el agente, comenzando la interacción en dicho estado, dependiendo de la política ejecutada por el agente.

Función de Estado - Valor para la Política π :

$$V^{\pi}(s) = E_{\pi}\{R_t | s_t = s\} = E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s\right\}$$

- El valor de la ejecución de una acción en un estado es el retorno esperado por el agente, comenzando la interacción en dicho estado a partir de la ejecución de dicha acción, dependiendo de la política ejecutada por el agente.

Función de Acción - Valor para la Política π :

$$Q^{\pi}(s,a) = E_{\pi}\{R_t | s_t = s, a_t = a\} = E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s, a_t = a\right\}$$

Una propiedad fundamental de las funciones de valor es que satisfacen ciertas propiedades recursivas.
Para cualquier política y cualquier estado s, V(s) y Q(s,a) pueden ser definidas recursivamente en términos de la denominada *Ecuación de Bellman* **(Bellman, 1957)**

1.10.2 Ecuación de Bellman

- La idea básica es:

$$\begin{aligned} R_t &= r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \gamma^3 r_{t+4} \cdots \\ &= r_{t+1} + \gamma\left(r_{t+2} + \gamma r_{t+3} + \gamma^2 r_{t+4} \cdots\right) \\ &= r_{t+1} + \gamma R_{t+1} \end{aligned}$$

- Entonces,

$$\begin{aligned} V^{\pi}(s) &= E_{\pi}\{R_t | s_t = s\} \\ &= E_{\pi}\{r_{t+1} + \gamma V^{\pi}(s_{t+1}) | s_t = s\} \end{aligned}$$

- O, sin el operador de valor esperado:

$$V^{\pi}(s) = \sum_a \pi(s,a) \sum_{s'} P^a_{ss'}\left[R^a_{ss'} + \gamma V^{\pi}(s')\right]$$

La ecuación anterior refleja el hecho de que el valor de un estado se encuentra definido en términos de la recompensa inmediata y los valores de los estados siguientes ponderados en función de las probabilidades de transición, y adicionalmente un factor de descuento.

1.10.3 Ecuación de Optimalidad de Bellman

La Ecuación de Optimalidad de Bellman refleja el hecho de que el Valor de un estado bajo la política óptima debe ser igual al retorno esperado para la mejor acción en dicho estado:

$$V^*(s) = \max_{a \in A} \sum_{s' \in S} \boldsymbol{P}(s,a,s')\left(R(s,a,s') + \gamma V^*(s')\right)$$

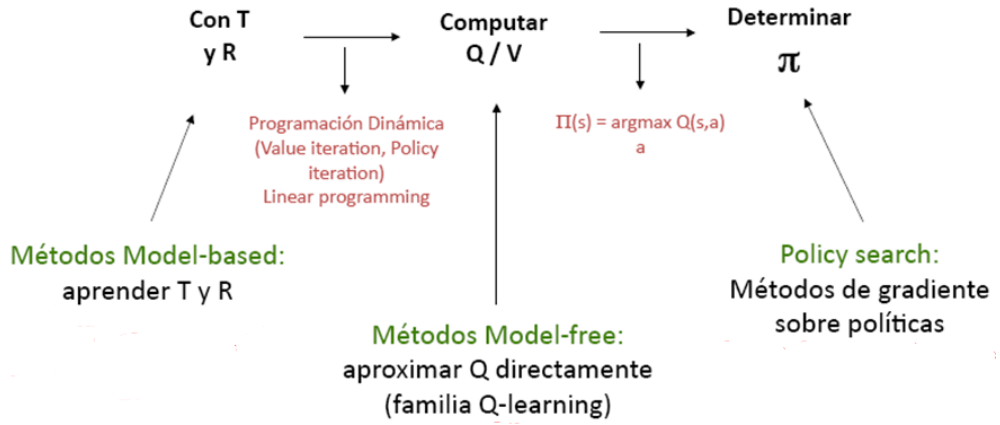
Al mismo tiempo, la acción óptima para un estado s dada la función de valor, puede obtenerse mediante:

$$\pi^*(s) = \arg\max_a \sum_{s' \in S} \boldsymbol{P}(s,a,s') \Big(R(s,a,s') + \gamma V^*(s') \Big)$$

La política anterior se denomina **Política Greedy**, dado que selecciona la mejor acción para cada estado, teniendo en cuenta la función de valor V(s).
De manera análoga, la función de acción-valor óptima puede expresarse como:

$$Q^*(s,a) = \sum_{s'} \boldsymbol{P}(s,a,s') \Big(R(s,a,s') + \gamma \max_{a'} Q^*(s',a') \Big)$$

1.11 Aproximaciones para el aprendizaje de V y Q mediante Soluciones Tabulares



1.11.1 Model Based vs. Model Free

- Model-free aprende Q/V directamente y presenta muy baja complejidad computacional.
- Model-based aprende T y R y usa un algoritmo de planning para encontrar la política. Uso eficiente de los datos/experiencia. Alto costo computacional.

1.11.2 Programación Dinámica: Iteración de Valor e Iteración de Política (Model Based)

Iteración de Valor

Value Iteration, for estimating $\pi \approx \pi_*$

Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation
Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(terminal) = 0$

Loop:
| $\Delta \leftarrow 0$
| Loop for each $s \in \mathcal{S}$:
| $v \leftarrow V(s)$
| $V(s) \leftarrow \max_a \sum_{s',r} p(s', r|s, a)\left[r + \gamma V(s')\right]$
| $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$

Output a deterministic policy, $\pi \approx \pi_*$, such that
$\pi(s) = \text{argmax}_a \sum_{s',r} p(s', r|s, a)\left[r + \gamma V(s')\right]$

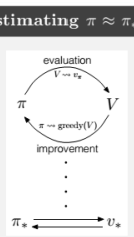
Iteración de Política

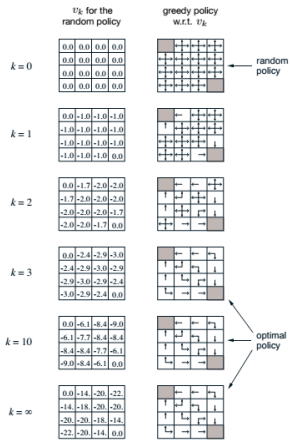
Policy Iteration (using iterative policy evaluation) for estimating $\pi \approx \pi_*$

1. Initialization
   $V(s) \in \mathbb{R}$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$

2. Policy Evaluation
   Loop:
   $\Delta \leftarrow 0$
   Loop for each $s \in \mathcal{S}$:
   $v \leftarrow V(s)$
   $V(s) \leftarrow \sum_{s',r} p(s', r|s, \pi(s))\left[r + \gamma V(s')\right]$
   $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
   until $\Delta < \theta$ (a small positive number determining the accuracy of estimation)

3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
   old-action $\leftarrow \pi(s)$
   $\pi(s) \leftarrow \text{argmax}_a \sum_{s',r} p(s', r|s, a)\left[r + \gamma V(s')\right]$
   If *old-action* $\neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*
   If *policy-stable*, then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2





In [ ]: