

Aprendizaje por Refuerzos

Teoría y Aplicaciones en Robótica,
Neurociencia y Psicología

Carlos “Greg” Diuk

Department of Psychology

Princeton Neuroscience Institute

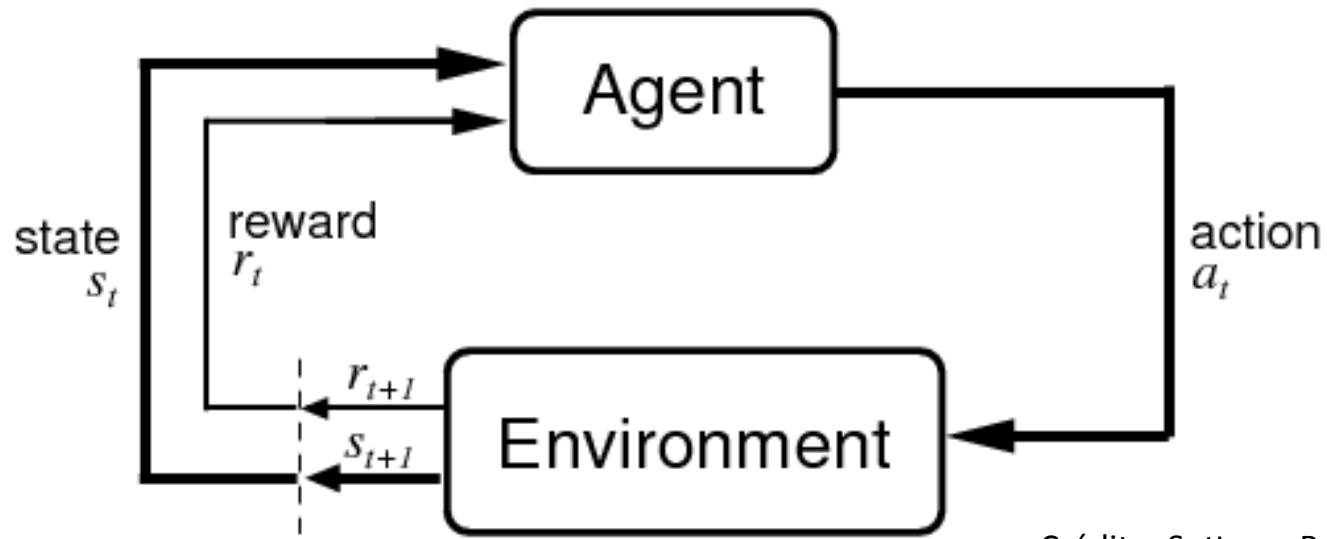
Princeton University



Resumen final

- Estudiamos el problema de la toma secuencial de decisiones: involucra aprender sobre nuestro entorno y elegir acciones que maximizan el retorno esperado.
- Estudios en aprendizaje animal muestran cómo los animales pueden aprender secuencias de acciones arbitrarias solamente maximizando refuerzos recibidos.
- Vimos condicionamiento Pavloviano e instrumental.
- RL computacional, inspirado por estas ideas, las formalizó y produjo un impacto importante en robótica, machine learning y neurociencias.

RL: el protocolo de interacción



Crédito: Sutton y Barto

El objetivo del agente es adaptar su comportamiento de manera de maximizar la suma de los refuerzos que espera obtener en el futuro.

El dilema exploración-explotación

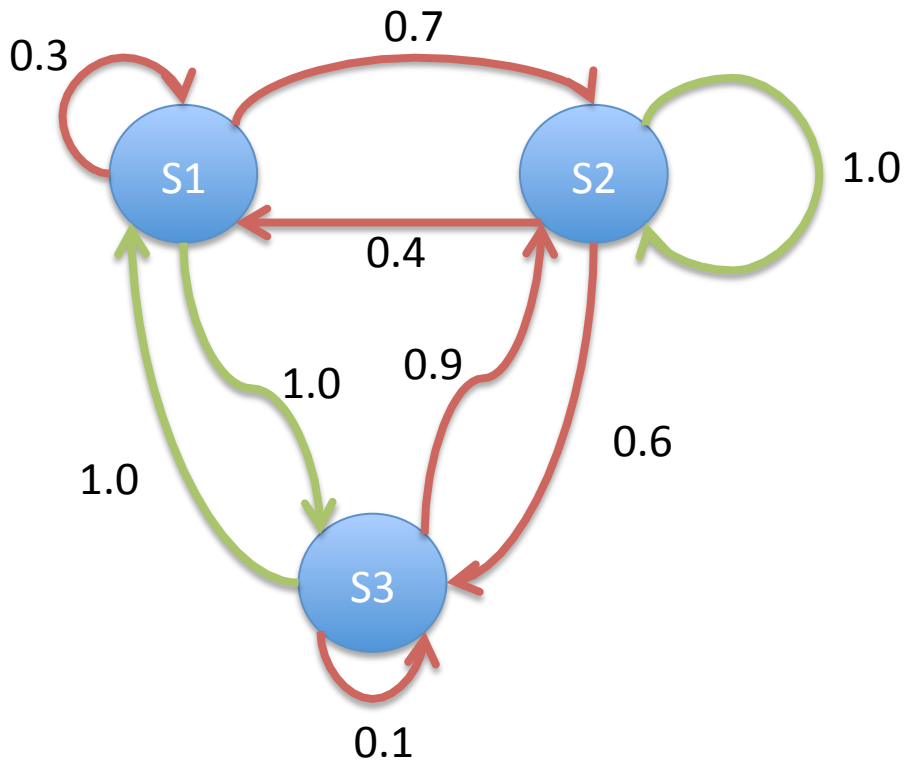
- Vimos el problema de los bandidos de k-brazos como un caso simple donde estudiar este dilema.
- Estrategias de Exp-Exp cuasi-óptimas basadas en optimismo ante la incertidumbre.

Procesos de Decisión de Markov

Markov Decision Process (MDP) en inglés:

- Conjunto de estados S , conjunto de acciones A
- Función de transición: $Pr(s'|s,a) = T(s,a,s')$.
- Función de refuerzo: $E[r|s,a] = R(s,a)$.
- Factor de descuento: γ

Ejemplo de MDP



$R(s1) = +1$

$R(s2) = 0$

$R(s3) = -1$

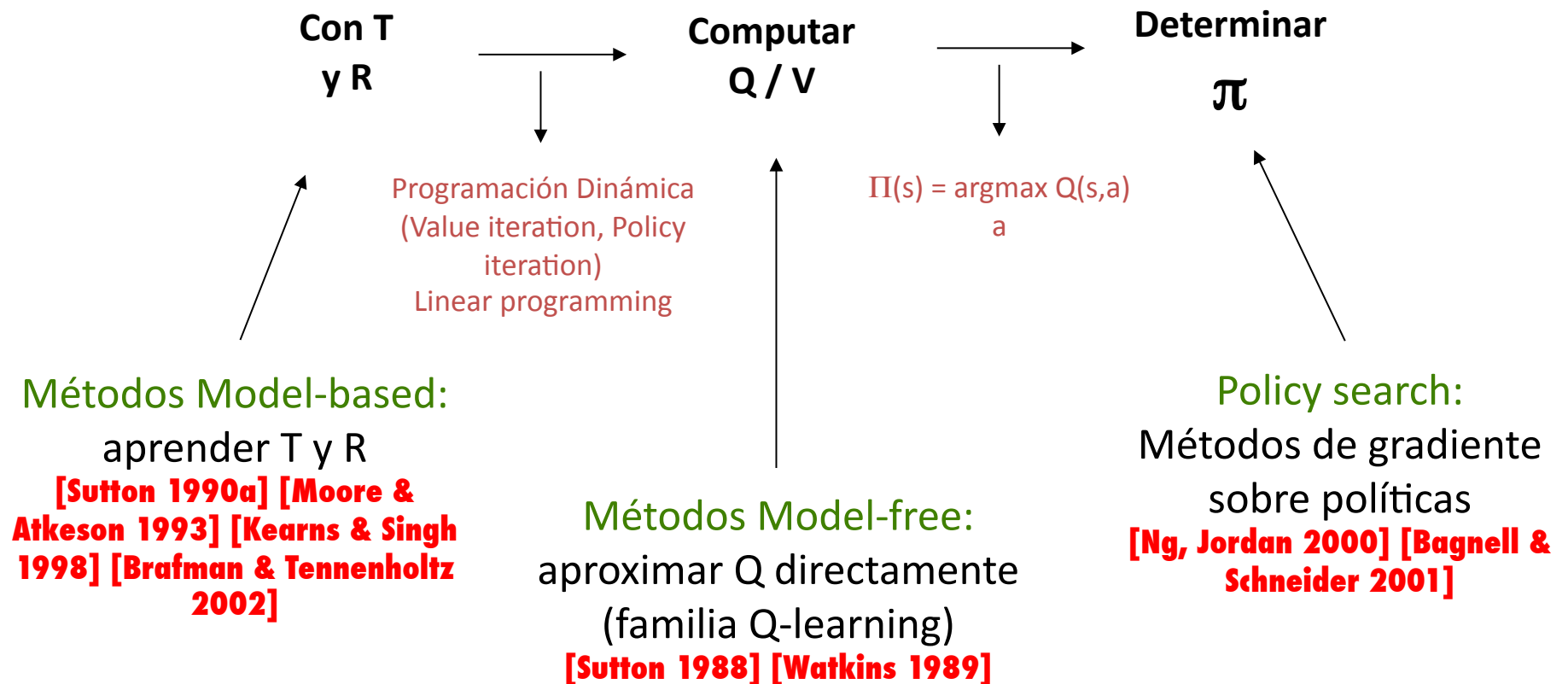
Función de transición T:

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

Planning

- Si conocemos T y R , podemos *resolver* el MDP de forma exacta (calcular V^*) usando value iteration, policy iteration o programación lineal.
- Los mismos métodos los podemos usar si tenemos un modelo de T y R .
- También vimos métodos de forward-planning: Sparse Sampling y UCT (Monte-Carlo Tree Search)

Aprendizaje



Tres fuentes de complejidad en RL

- **Experiencia (Sample / Learning)**
 - Cantidad de experiencia/interacción con el mundo real necesaria por un algoritmo para alcanzar performance cuasi-óptima. [Kakade 2003]
 - Puede ser crítica en robótica!
- **Computacional (Planning)**
 - Cantidad de esfuerzo computacional necesario por cada experiencia adquirida.
- **Espacio**
 - Espacio necesario para almacenar experiencia y estructuras de datos.

Model-based

- Algoritmos **model-based** construyen modelos de T y R (R_{\max} y familia, DYNA)
 - Se puede hacer exploración inteligente: R_{\max} y el optimismo ante la incertidumbre.
 - Uso eficiente de la experiencia (bajo sample complexity).
 - Alto costo computacional porque hay que correr un algoritmo de planning para resolver el modelo (VI u otro).

Model-free y Policy Search

- Algoritmos **model-free** aproximan Q/V directamente.
 - Menos oportunidades de exploración inteligente.
 - Bajo costo computacional.
- **Policy search** usa métodos de gradiente para encontrar políticas directamente.
- Aplicaciones: caminata rápida en AIBo y Helicopter.

Model-free II

- Algoritmos model-free:
 - Vimos que Q-learning no tiene garantías de convergencia polinomial.
 - SARSA(λ): actualiza varios estados, según cuándo y con qué frecuencia fueron visitados. Mejora en la práctica, pero hay casos en que es malo.
 - Replay de experiencia: también ayuda a veces en la práctica.

RL en el cerebro

- Model-free en el cerebro:
 - Señal de dopamina en ventral striatum responde a errores de predicción.
- Model-based en el cerebro:
 - Un poco más complicado y se sabe menos, pero hay modelos. DLPFC codifica valor? OFC representaciones del estado?

Un cerebro, dos sistemas?

- Los estudios en lesiones lo sugieren.
- Cada sistema es útil en distinto momento:
 - **Goal-directed / Model-based**: buena cuando tenemos poca experiencia (uso eficiente de los datos) y el refuerzo “está cerca” (el problema de planning no es muy difícil)
 - **Habitual / Model-free**: bueno después de mucho entrenamiento.
- Arbitración: necesidad de usar el sistema más confiable.

Representaciones

- Representaciones factorizadas:
 - Representando dependencias con DBNs.
 - Factored- R_{\max} : DBNs provistos en el input.
 - Los k-meteorólogos: aplicación a aprender la estructura de los DBNs.
- Representación basada en objetos:
 - Representación basada en objetos y sus relaciones. Reglas condición-efecto.
 - DOORmax para el caso de condiciones-efectos determinísticos.

Espacios continuos

- Si el conjunto de estados o de acciones es continuo, ya no podemos usar tablas.
- Hay que usar algún método de aproximación (ídem si el número de estados es muy grande, aunque sea discreto).
- Para resolver un MDP continuo se puede usar fitted V o fitted Q iteration, o forward planning.
- Vimos un par de métodos model-based: MPA y MRE.

Algoritmos:

Pseudo-código y/o referencias

Q-learning

Algorithm 2: Q-learning

Input: α, γ

Initialize $Q(s, a)$ according to initialization policy.

for all timestep $t = 1, 2, 3, \dots$ do

 Observe current state s_t

 Choose action a_t according to exploration policy

 Execute action a_t , obtain reward r_t and observe new state s_{t+1} .

 Set α according to learning rate polict.

 Update Q : $Q(s, a) \leftarrow (1 - \alpha)Q(s, a) + \alpha[r + \gamma \max_{a'} Q(s', a')]$

end for

SARSA(λ)

Initialize $Q(s, a)$ arbitrarily and $e(s, a) = 0$, for all s, a

Repeat (for each episode):

Initialize s, a

Repeat (for each step of episode):

Take action a , observe r, s'

Choose a' from s' using policy derived from Q (e.g., ϵ -greedy)

$\delta \leftarrow r + \gamma Q(s', a') - Q(s, a)$

$e(s, a) \leftarrow e(s, a) + 1$

For all s, a :

$Q(s, a) \leftarrow Q(s, a) + \alpha \delta e(s, a)$

$e(s, a) \leftarrow \gamma \lambda e(s, a)$

$s \leftarrow s'; a \leftarrow a'$

until s is terminal

DYNA

Sutton, R.S. (1990). Integrated architectures for learning, planning, and reacting based on approximating dynamic programming.
International Conference on Machine Learning
(ICML 1990)

<http://webdocs.cs.ualberta.ca/~sutton/publications.html#Dyna>

MAXQ

Dietterich, T. G. (2000). Hierarchical reinforcement learning with the MAXQ value function decomposition. *Journal of Artificial Intelligence Research*, 13, 227–303.

DSHP

Diuk, C., Littman, M., & Strehl, A. (2006). A hierarchical approach to efficient reinforcement learning in deterministic domains. *Fifth International Conference on Autonomous Agents and Multiagent Systems (AAMAS-06)*.

R-Max (1)

Brafman, R. I., & Tenenbholz, M. (2002). R-MAX —a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3, 213–231.

- Muy difícil de leer! Pero el algoritmo es muy sencillo (ver próximos slides).

R-Max (2)

Algorithm 1: KWIK- R_{\max}

```
1: Input:  $\mathcal{S}, \mathcal{A}, \gamma, A_T, A_R, \epsilon_T, \epsilon_R, \delta_T, \delta_R, \epsilon_P$ .
2: Initialize  $A_T$  with parameters  $\epsilon_T$  and  $\delta_T$ .
3: Initialize  $A_R$  with parameters  $\epsilon_R$  and  $\delta_R$ .
4: for all timesteps  $t = 1, 2, 3, \dots$  do
5:   Update empirical known state-action MDP  $\hat{M} = \langle \mathcal{S}, \mathcal{A}, \hat{T}, \hat{R}, \gamma \rangle$ :
6:   for all  $(s, a) \in \mathcal{S} \times \mathcal{A}$  do
7:     if  $A_T(s, a) = \perp$  or  $A_R(s, a) = \perp$  then
8:       
$$\hat{T}(s, a, s') = \begin{cases} 1 & \text{if } s' = s \\ 0 & \text{otherwise} \end{cases}$$

9:        $\hat{R}(s, a) = r_{\max}$ 
10:     else
11:        $\hat{T}(s, a, s') = A_T(s, a)$ 
12:        $\hat{R}(s, a) = A_R(s, a)$ 
13:     end if
14:   end for
15:   Compute a near-optimal value function  $Q_t$  for  $\hat{M}$ .
16:   Observe current state  $s_t$ , greedily choose action  $a_t = \operatorname{argmax}_{a \in \mathcal{A}} Q_t(s, a)$ , receive
   reward  $r_t$  and transition to  $s_{t+1}$ .
17:   if  $A_T(s_t, a_t) = \perp$  then
18:     Provide  $A_T$  with the sample  $\langle s_t, a_t, s_{t+1} \rangle$ 
19:   end if
20:   if  $A_R(s_t, a_t) = \perp$  then
21:     Provide  $A_R$  with the sample  $\langle s_t, a_t, r_t \rangle$ 
22:   end if
23: end for
```

R-Max (3)

Algorithm 3: R_{\max}^T AddExperience

Input: s, a, s'

Update transition count: $n(s, a, s') \leftarrow n(s, a, s') + 1$.

Increase count $n(s, a) \leftarrow n(s, a) + 1$.

Update empirical transition distribution: $\hat{T}(s, a, s') \leftarrow n(s, a, s')/n(s, a)$

Algorithm 4: R_{\max}^R AddExperience

Input: s, a, r

Update empirical total reward $r(s, a) \leftarrow r(s, a) + r$.

Increase count $n(s, a) \leftarrow n(s, a) + 1$.

Update empirical reward distribution: $\hat{R}(s, a) \leftarrow \frac{r(s, a)}{n(s, a)}$

Factored R-Max

Guestrin, C., Patrascu, R., & Schuurmans, D. (2002). Algorithm-directed exploration for model-based reinforcement learning in factored MDPs. *Proceedings of the International Conference on Machine Learning* (pp. 235–242).

Met-Rmax

Diuk, C., Li, L., & Leffler, B. R. (2009). The adaptive *k-meteorologists problem* and its application to structure learning and feature selection in reinforcement learning. *Proceedings of the Twenty-Sixth International Conference on Machine Learning (ICML-09)*.

DOORMAX

Diuk, C., Cohen, A., & Littman, M. L. (2008). An object-oriented representation for efficient reinforcement learning. *Proceedings of the Twenty-Fifth International Conference (ICML 2008)*.

Referencias



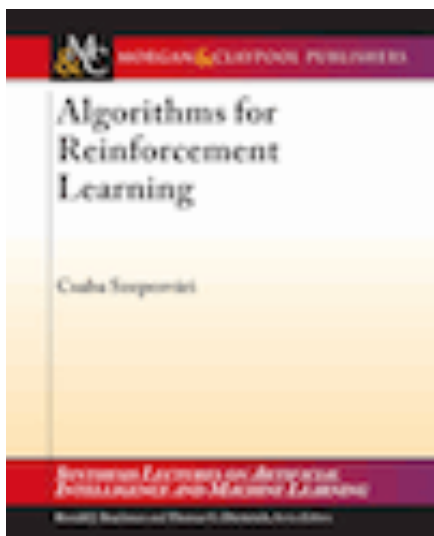
Richard S. Sutton and Andrew G. Barto

“El” libro, de Sutton y Barto.

Está disponible online en formato HTML:

<http://www.cs.ualberta.ca/~sutton/book/ebook/the-book.html>

Referencias



Libro de Csaba Szepesvári.

Recién salido del horno (menos de 1 mes).
Bastante teórico pero con muchos algoritmos y resultados bien actualizados.

Está disponible online en PDF:

[http://www.sztaki.hu/~szcsaba/papers/
RLAlgsInMDPs-lecture.pdf](http://www.sztaki.hu/~szcsaba/papers/RLAlgsInMDPs-lecture.pdf)

Recursos

- Lista de RL: <http://groups.google.com/group/rl-list>
- Página de Satinder Singh con referencias a algoritmos, mitos de RL, demos, etc: <http://umichrl.pbworks.com/Algorithms-of-Reinforcement-Learning>
- RL Competition:
<http://2009.rl-competition.org/getstarted.php>
- RL Glue:
http://glue.rl-community.org/wiki/Main_Page
- RL Library:
http://library.rl-community.org/wiki/Main_Page