

The background features a robotic arm with glowing blue circular elements at its joints, reaching towards a stylized brain composed of a circuit board on the left and organic, colorful lines on the right. A human hand is shown pointing at the brain. The overall theme is the intersection of robotics and artificial intelligence.

Inteligencia Artificial

Clase 24: aprendizaje por refuerzo I



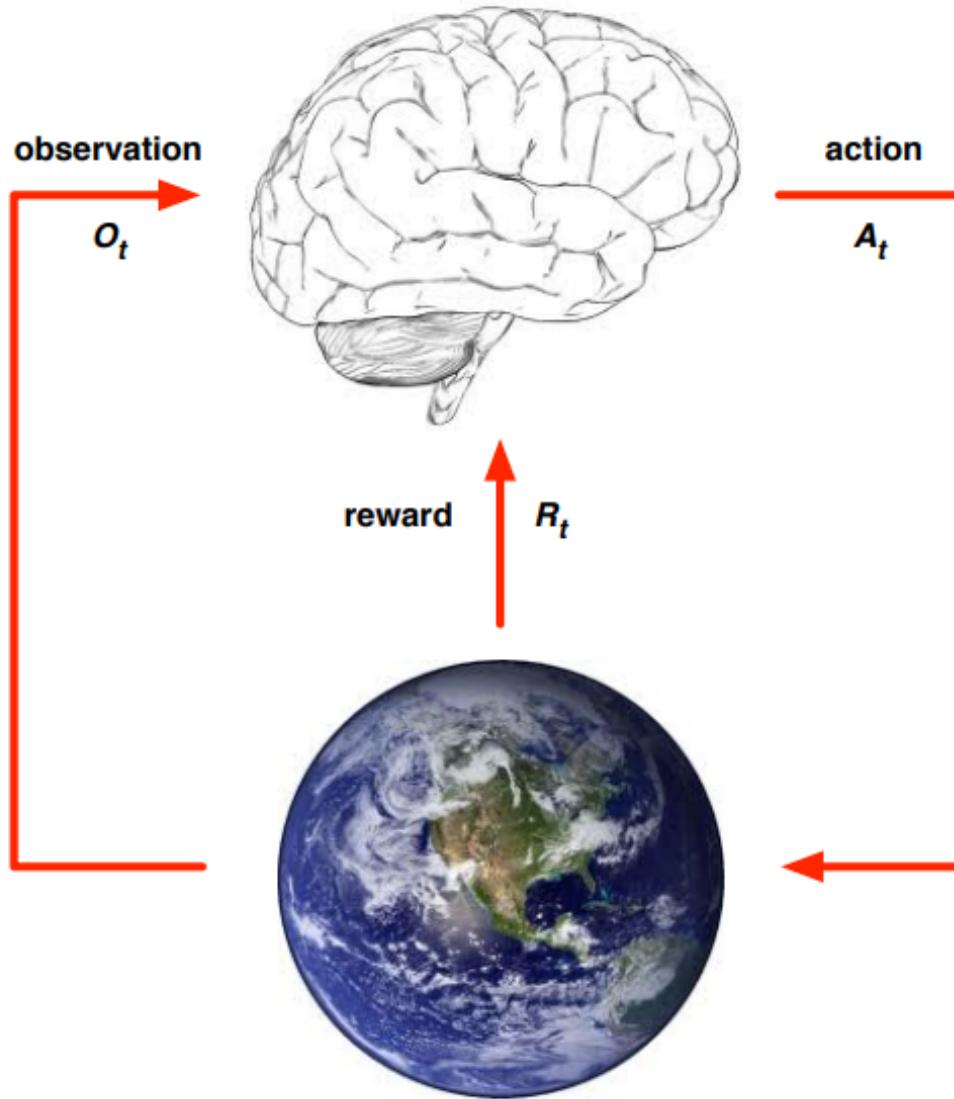
Para el día de hoy...

- Aprendizaje por refuerzo

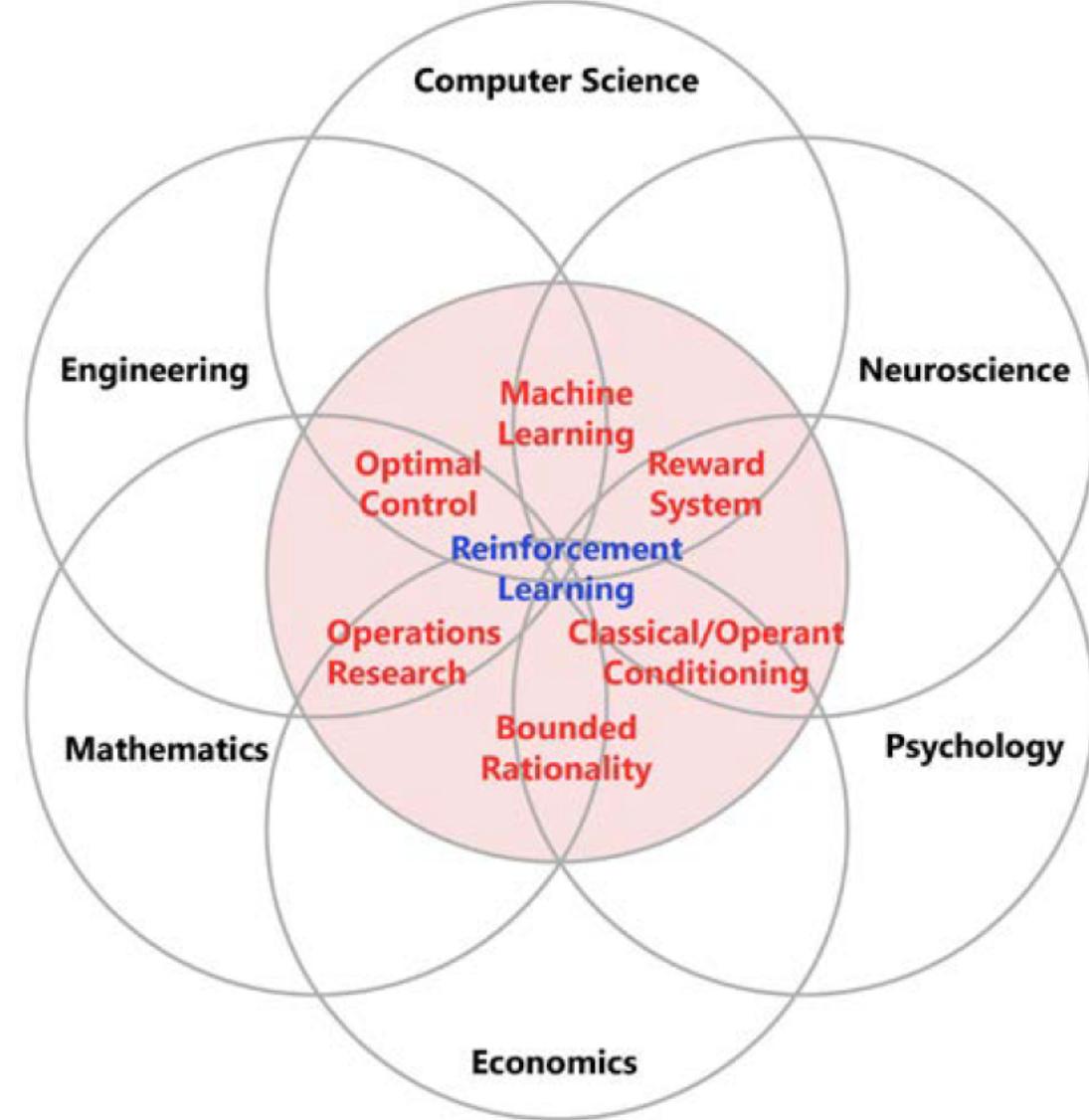


Aprendizaje por refuerzo

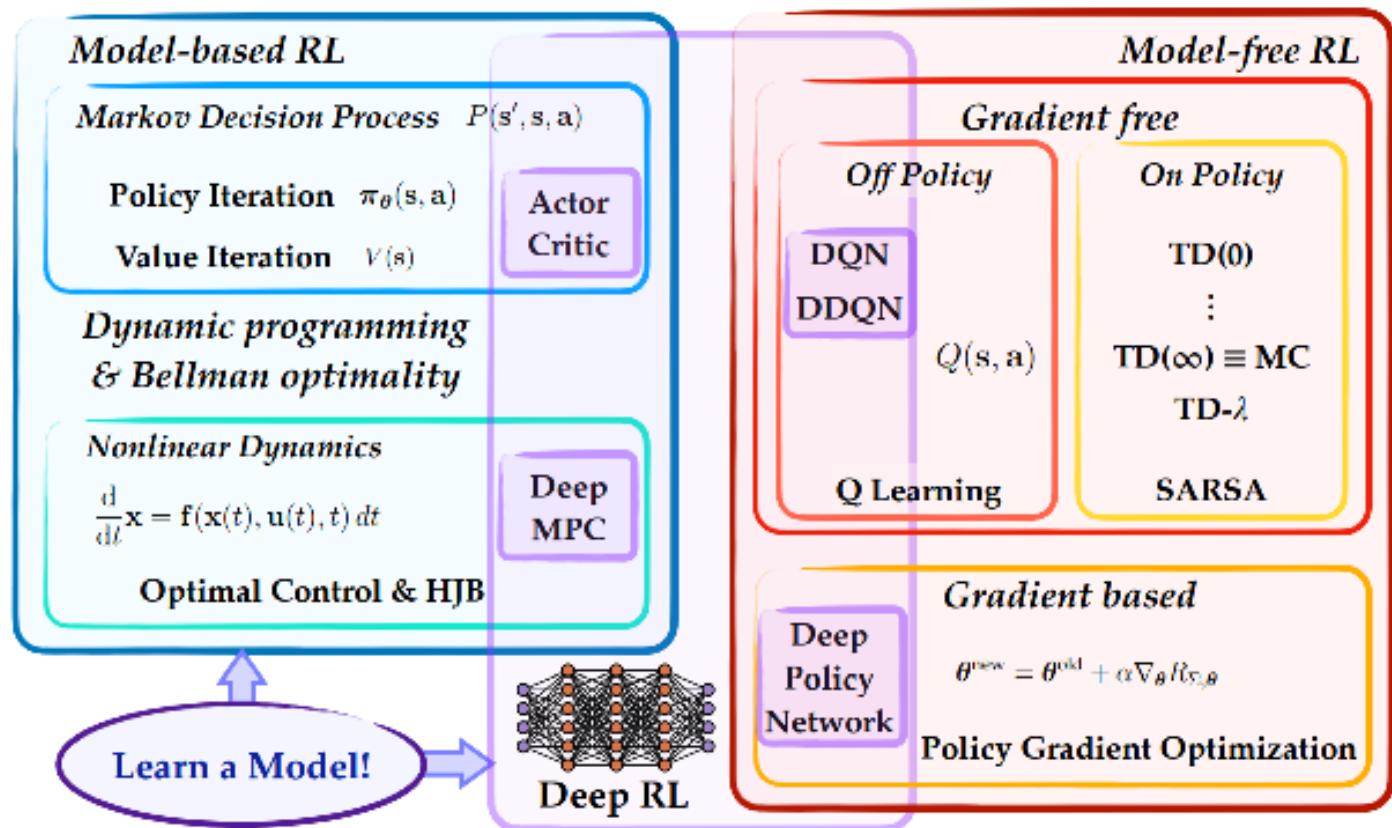
- Un agente recibe retroalimentación en la forma de recompensas
- La utilidad del agente está definida por una función de premio
- El agente debe actuar de forma que maximice el valor esperado de las recompensas
- Todo el aprendizaje está basado en los ejemplos observados



¿En dónde está ubicado?



Taxonomía de los métodos



El trabajo...

nature

Explore content ▾ About the journal ▾ Publish with us ▾ Subscribe

nature > letters > article

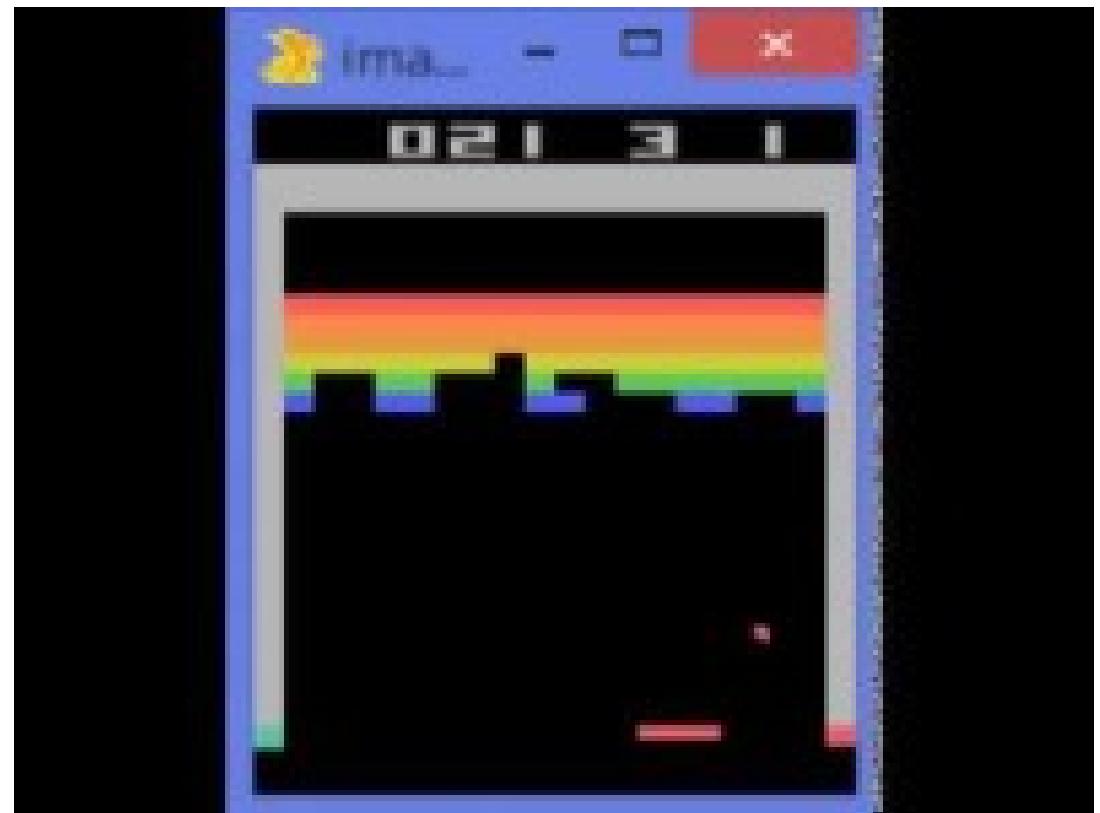
Published: 25 February 2015

Human-level control through deep reinforcement learning

Volodymyr Mnih, Koray Kavukcuoglu✉, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg & Demis Hassabis✉

Nature 518, 529–533 (2015) | Cite this article

397k Accesses | 7885 Citations | 1550 Altmetric | Metrics



Aplicaciones

Vehículos
autónomos

Automatización
en la industria

Trading
automático

Procesamiento
de lenguaje
natural

Juegos/vídeo
juegos

Robótica

Marketing

Colocación de
recursos

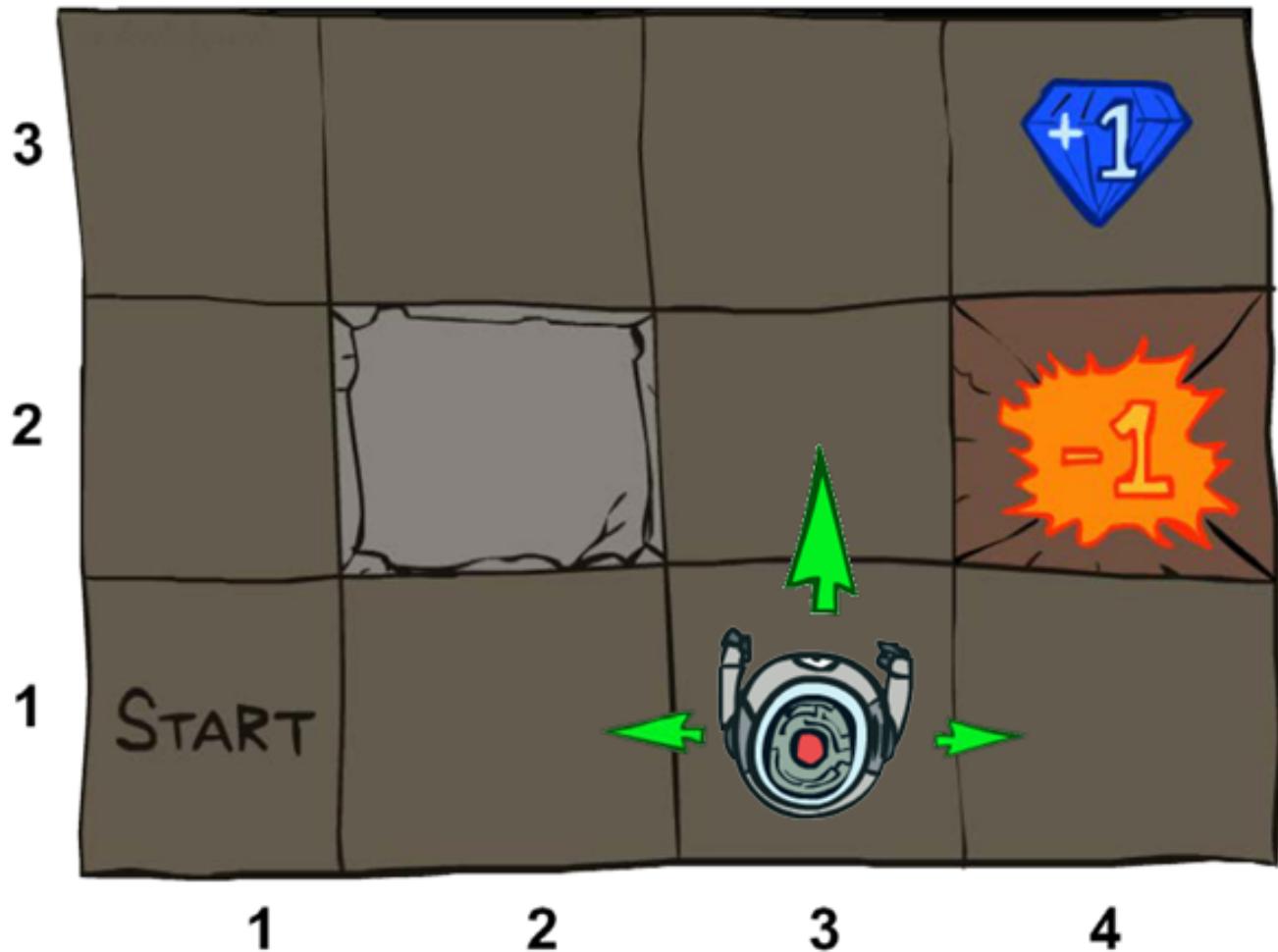
Control de
semáforos

Manufactura

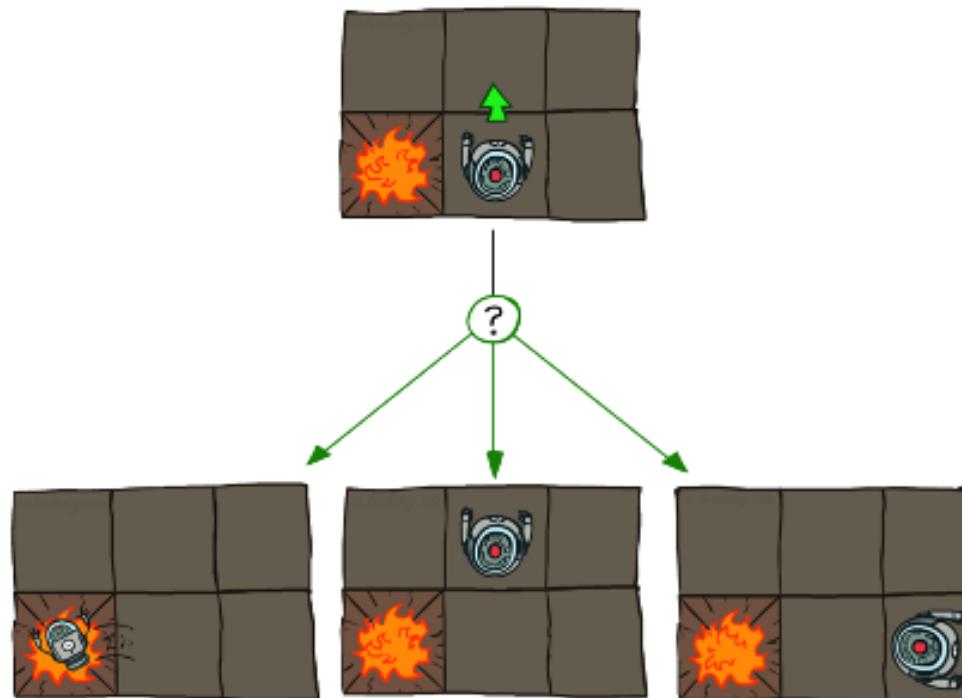
Balanceo de
cargas

Un ejemplo

- Un nuevo problema
 - El agente vive en una rejilla
 - Existen paredes que bloquean el camino
- Las acciones no siempre suceden
 - 80% del tiempo la acción “norte” lleva al agente al “norte”
 - 10% del tiempo la acción “norte” toma al agente al “oeste”
 - 10% del tiempo la acción “norte” toma al agente al “este”
 - Si existe una pared en la dirección del movimiento, el agente se queda donde estaba
- El agente recibe recompensa en cada paso
 - Una pequeña recompensa en cada paso (puede ser negativa)
 - Una recompensa grande al final (buena o mala)
- La meta: maximizar la suma de recompensas



Las acciones

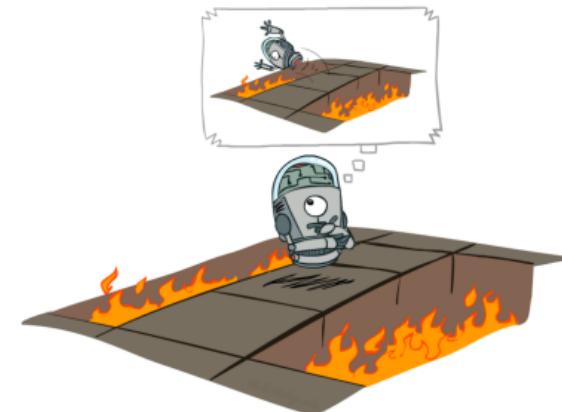


Procesos de decisión de Markov

- Los elementos
 - Un conjunto de estados $s \in S$
 - Un conjunto de acciones $a \in A$
 - Un modelo $T(s,a,s')$
 - Una función de recompensa $R(s,a,s')$
 - Un estado inicial
 - Tal vez un estado terminal
- La meta es encontrar una política $\pi^*: S \rightarrow A$
- En aprendizaje por refuerzo T o R son desconocidos



Online Learning



Offline Solution

¿Qué significa Markov?

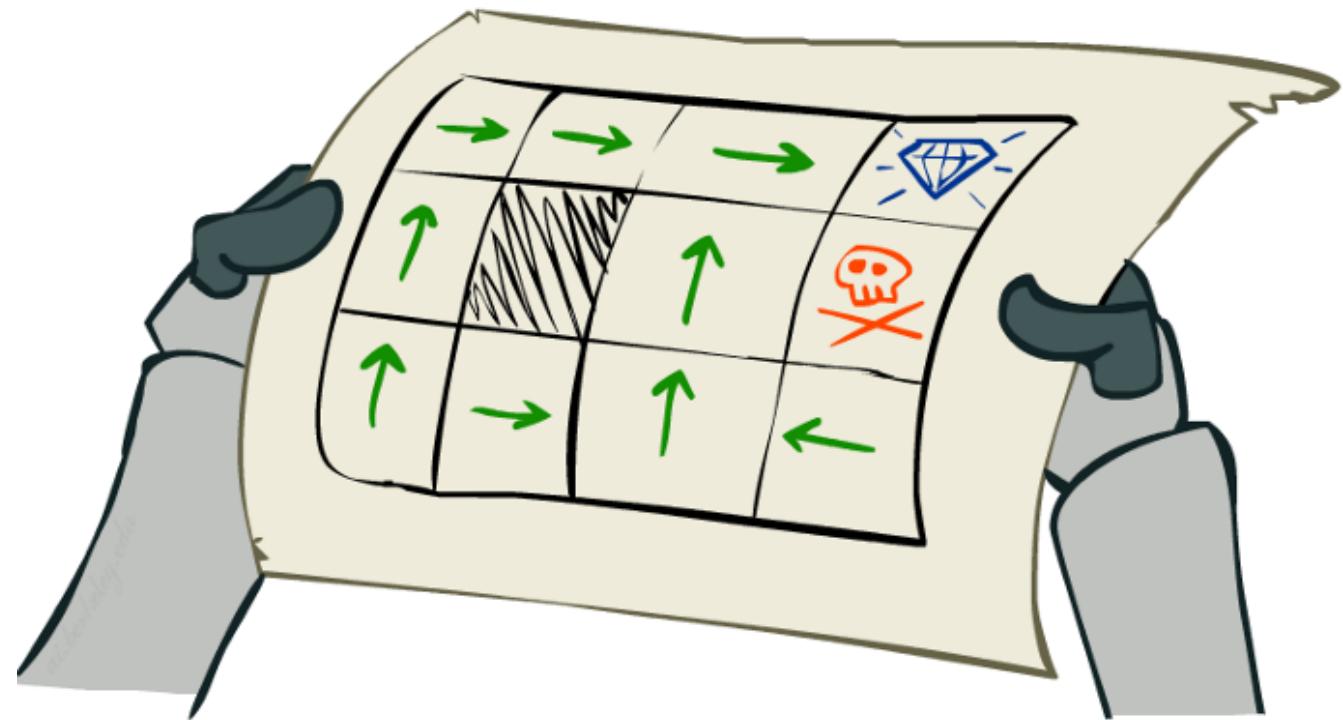
- Dado el presente, el futuro y el pasado son independientes
- En procesos de decisión de Markov eso significa
$$P(S_{t+1} = s' | S_t = s_t, A_t = a_t, \dots, S_0 = s_0) = P(S_{t+1} = s' | S_t = s_t, A_t = a_t)$$
- Eso quiere decir que la función solo depende del estado actual y no de la historia



Andrey Markov
(1856-1922)

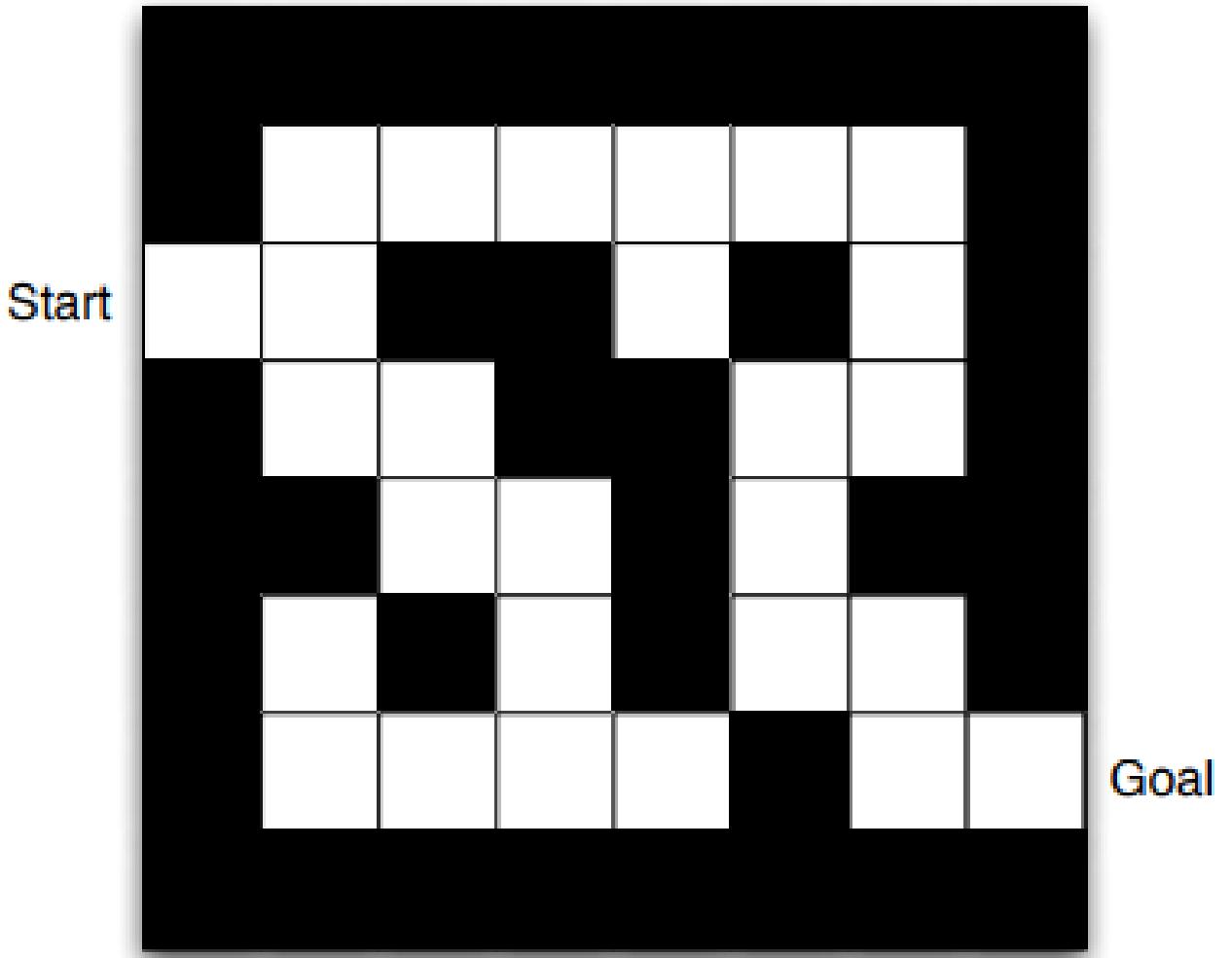
Política

- En los problemas mono agentes determinista de búsqueda, queríamos un plan óptimo o secuencia de acciones
- Para MDPs, queremos una política óptima $\pi^*: S \rightarrow A$
 - Una política π da una acción para cada estado
 - Una política óptima es una que maximiza la política esperada
 - Una política explícita define un agente de reflejo



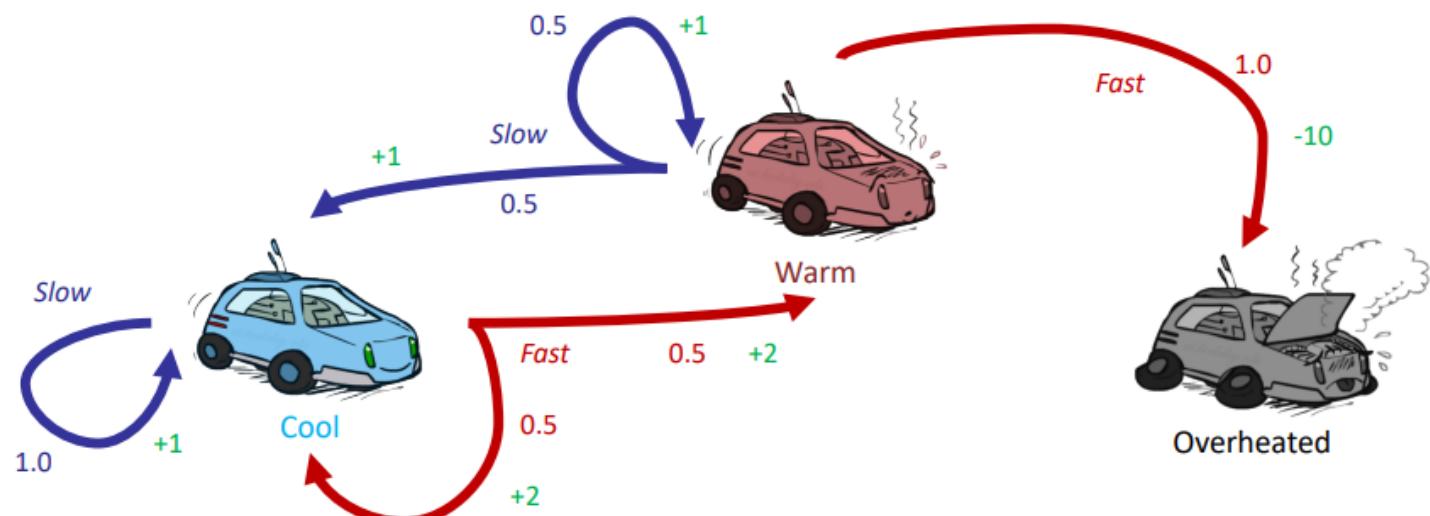
Un ejemplo

- Recompensas: -1 por paso
- Acciones: N, S, E, O
- Estados: ubicación del agente



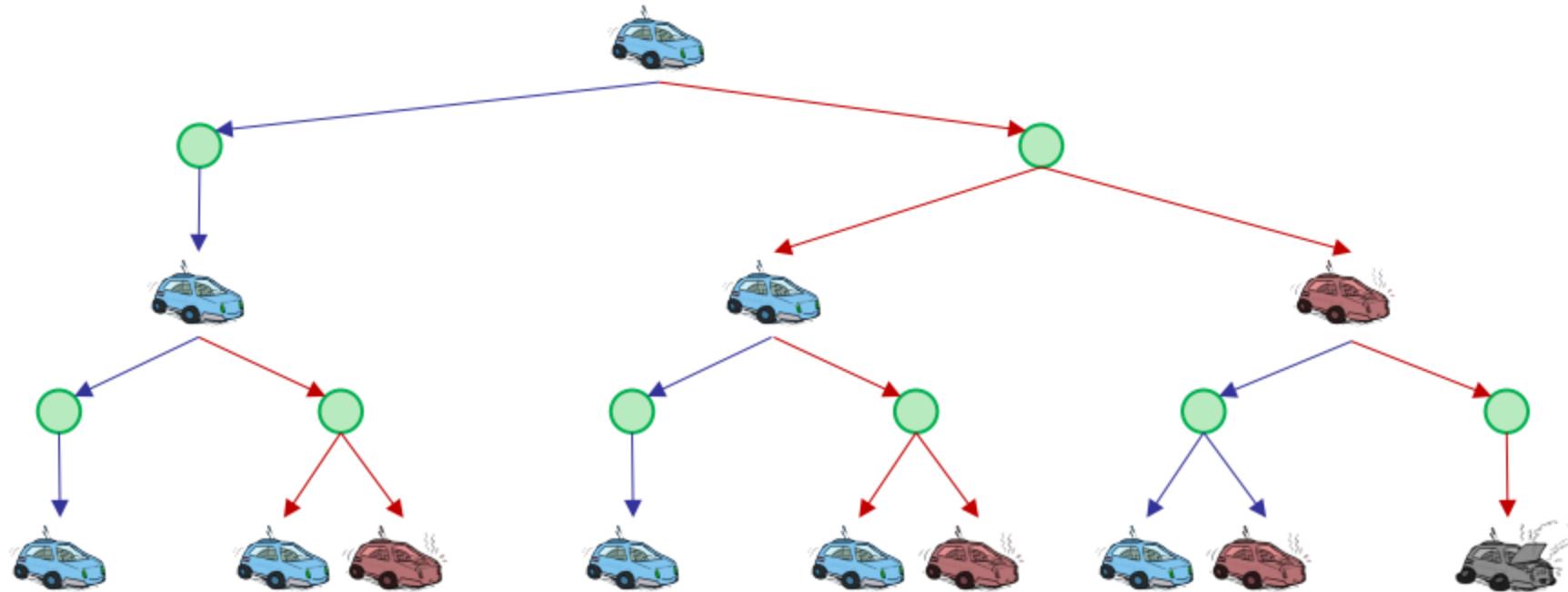
Un ejemplo: conduciendo un auto

- Un auto quiere viajar rápido
- Existen 3 estados: Cool, Warm, Overheated
- Existen 2 acciones: Slow, Fast
- Ir rápido significa obtener el doble de la recompensa



¿Qué hacemos?

¿Qué tal un árbol?



Secuencia de utilidades

¿Qué deberíamos preferir?

¿Más o menos?
[1,2,2] o [2,3,4]

Ahora o más tarde?
[0,0,1] o [1,0,0]

Descuento

- Es razonable maximizar la suma de recompensas
- También preferir recompensas ahora que después
- Una opción, las recompensas decaen exponencialmente



1



γ



γ^2

Ejemplo de descuentos

10					1
a	b	c	d	e	

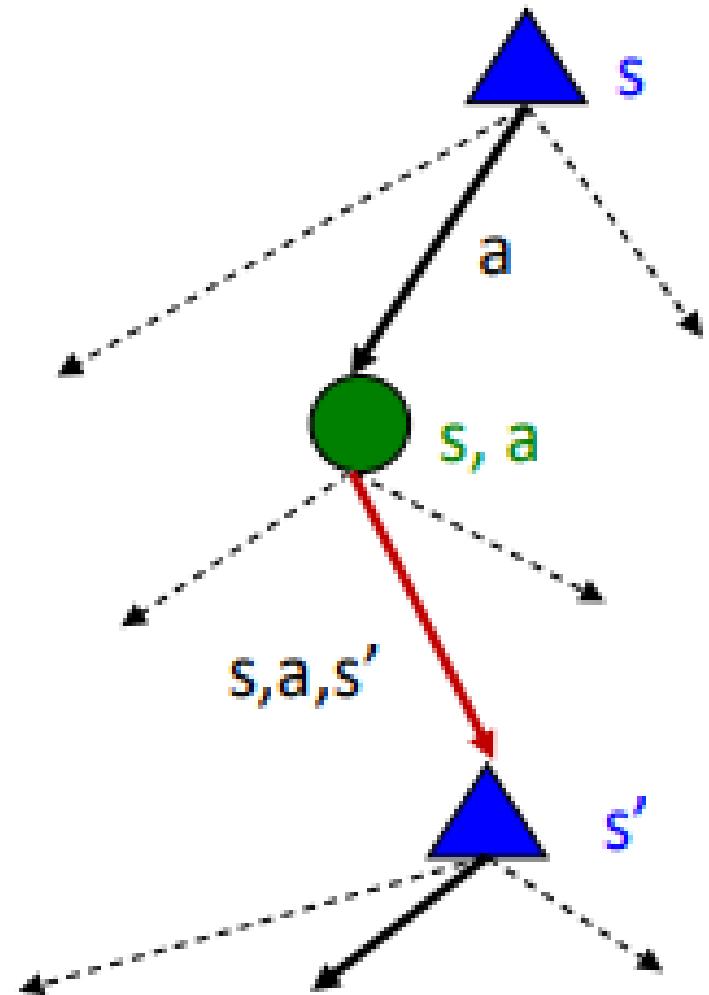
- Dado
 - Acciones: Este, Oeste y Salida (solo disponible en a, e)
 - Transiciones: deterministas
 - Pregunta 1: para $\gamma = 1$, cuál es la política óptima?
 - Pregunta 2: para $\gamma = 0.1$, cuál es la política óptima?
 - Pregunta 3: para cual γ Oeste y Este son igualmente de buenas iniciando en d

Utilidades infinitas

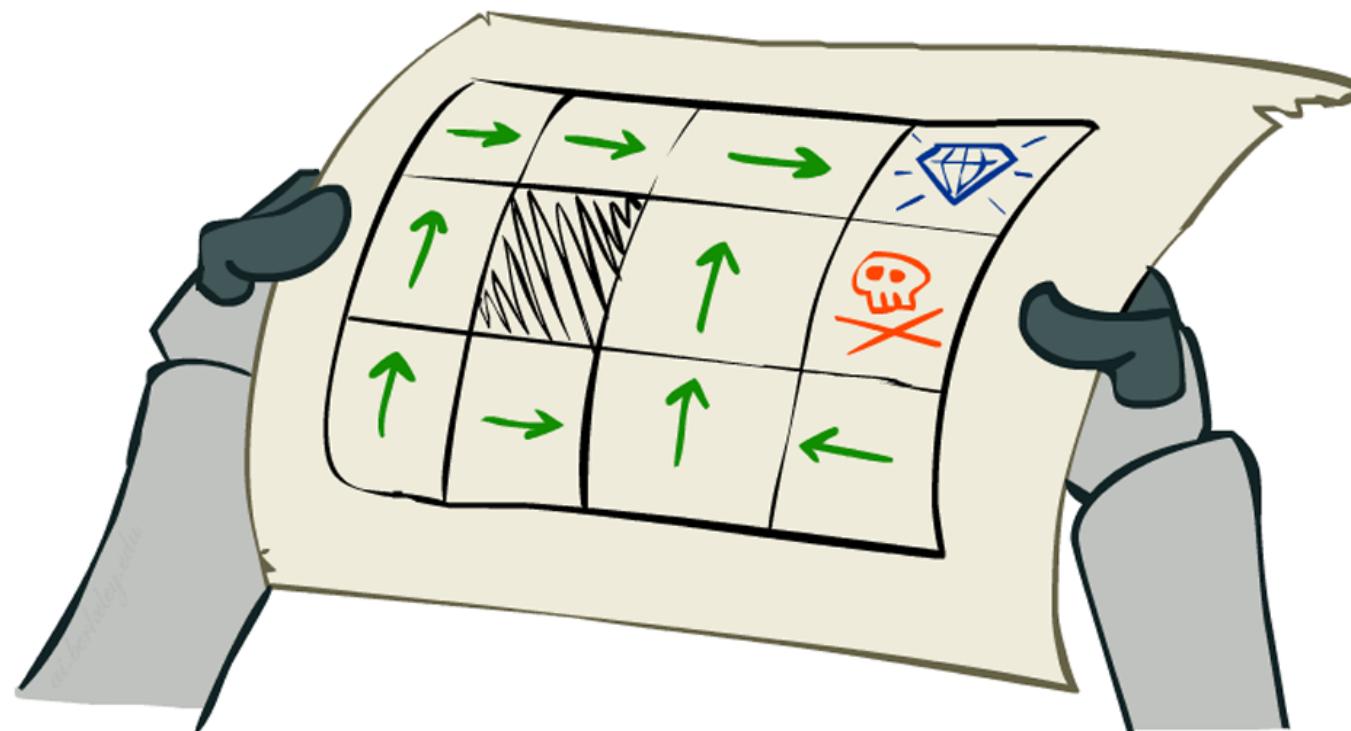
- ¿Qué pasa si el problema es infinito?
- Solución
 - Resolver para un horizonte finito
 - Descuento: $0 < \gamma < 1$
 - $$U([r_0, \dots, r_\infty]) = \sum_{t=0}^{\infty} \gamma^t r_t \leq \frac{R_{\max}}{1-\gamma}$$
 - Estado absorbente: garantiza que para cada política, eventualmente siempre se alcanza un estado terminal

Cantidades optimas

- El valor o utilidad de un estado s
 - $V^*(s)$ = utilidad esperada iniciando en s y actuando de forma óptima
- El valor o utilidad de un estado q
 - $Q^*(s, a)$ = utilidad esperada iniciando con la acción a desde el estado s y actuando de forma óptima posteriormente
- La política óptima
 - $\pi^*(s)$ = acción óptima desde el estado s

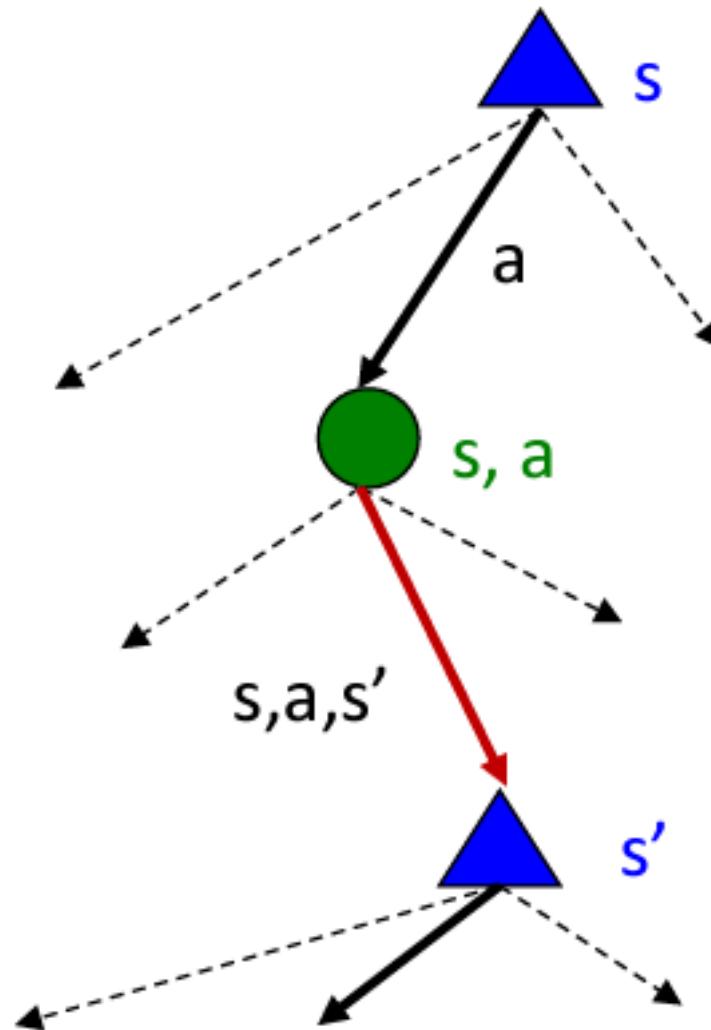


Resolviendo MDPs



Cantidad óptimas

- El valor de un estado s
 - $V^*(s)$ = utilidad esperada iniciando en s y actuando de forma óptima
- El valor de un estado $q(s, a)$
 - $Q^*(s, a)$ = utilidad esperada iniciando en una acción a desde el estado s y después actuando de forma óptima
- La política óptima
 - $\pi^*(s)$ = acción óptima iniciando en el estado s



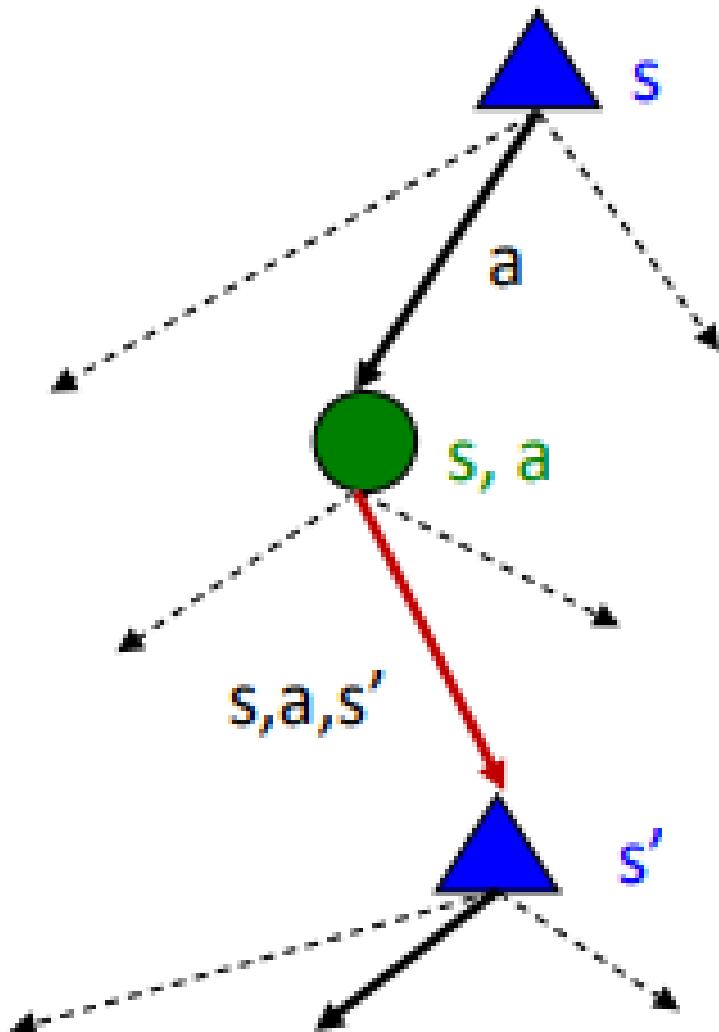
s is a
state

(s, a) is a
q-state

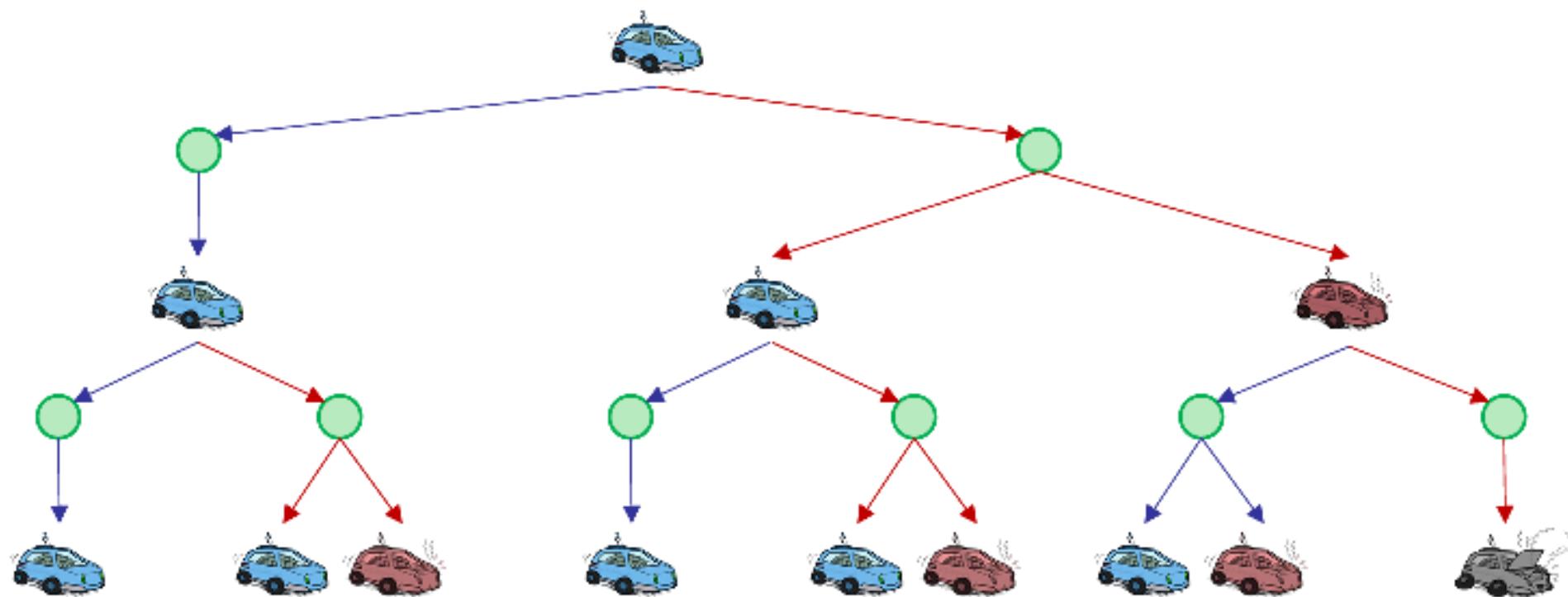
(s, a, s') is a
transition

¿Cómo lo resolvemos?

- Calcular el valor esperado de un estado (expectimax)
 - Utilidad esperada bajo la acción óptima
 - Media de la suma de las recompensas (descontadas)
- Definición recursiva del valor
 - $V^*(s) = \max_a Q^*(s, a)$
 - $Q^*(s, a) = \sum_{s'} T(s, a, s') [R(s, a, s') + \gamma V^*(s')]$
 - $V^*(s) = \max_a \sum_{s'} T(s, a, s') [R(s, a, s') + \gamma V^*(s')]$

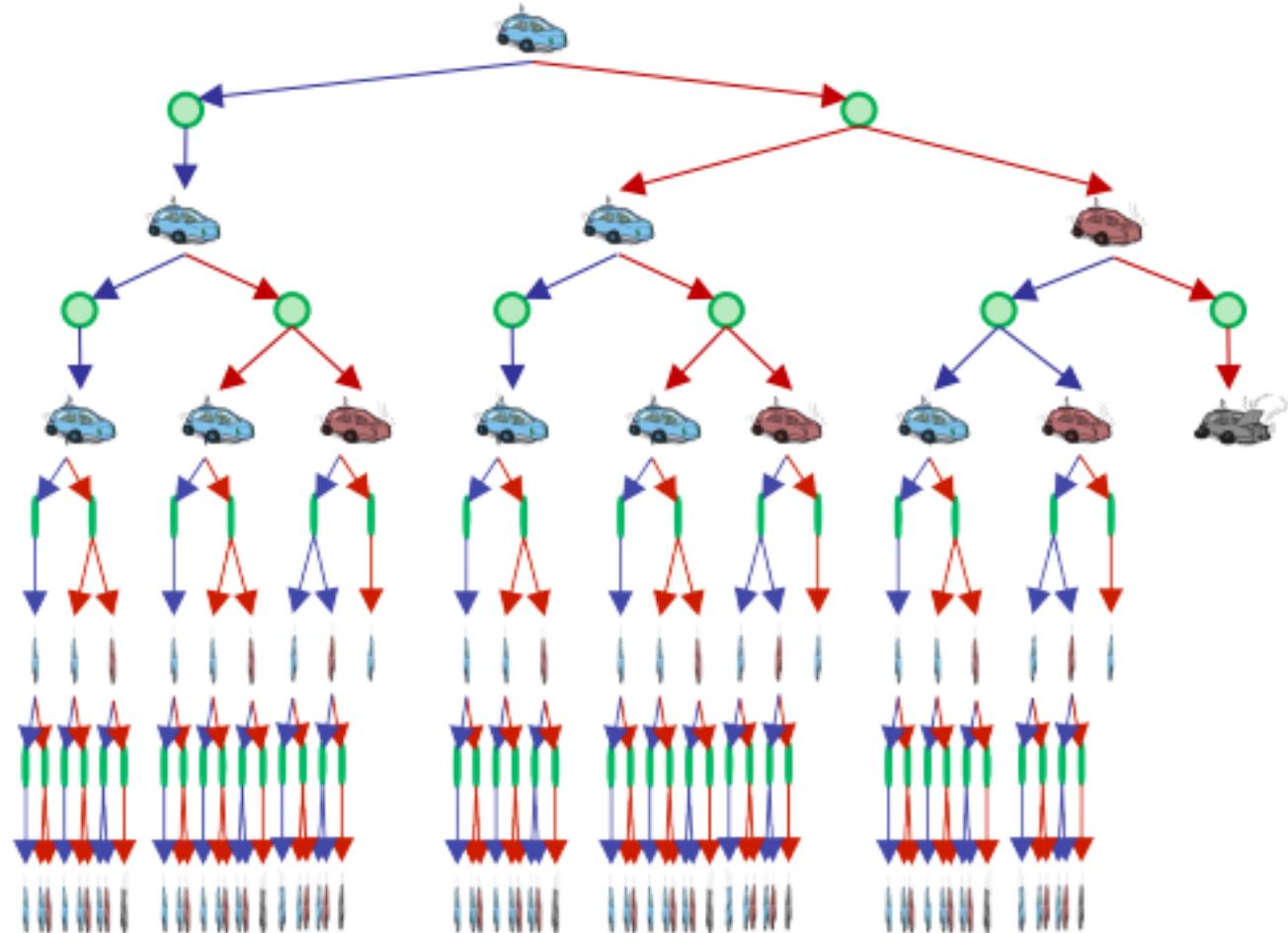


Regresando a nuestro ejemplo I



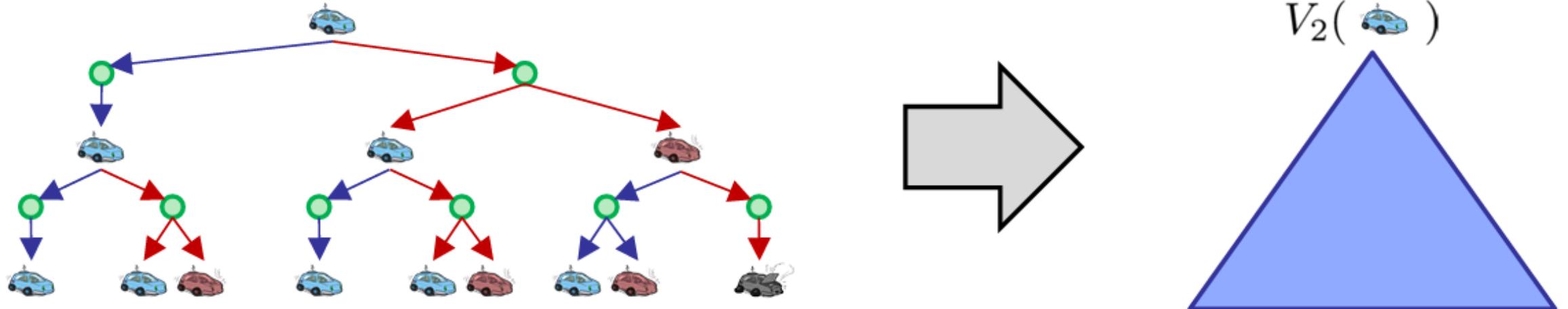
Regresando a nuestro ejemplo II

- Parece que estamos haciendo mucho trabajo
- Problemas
 - Los estados se repiten
 - El árbol sigue infinitamente

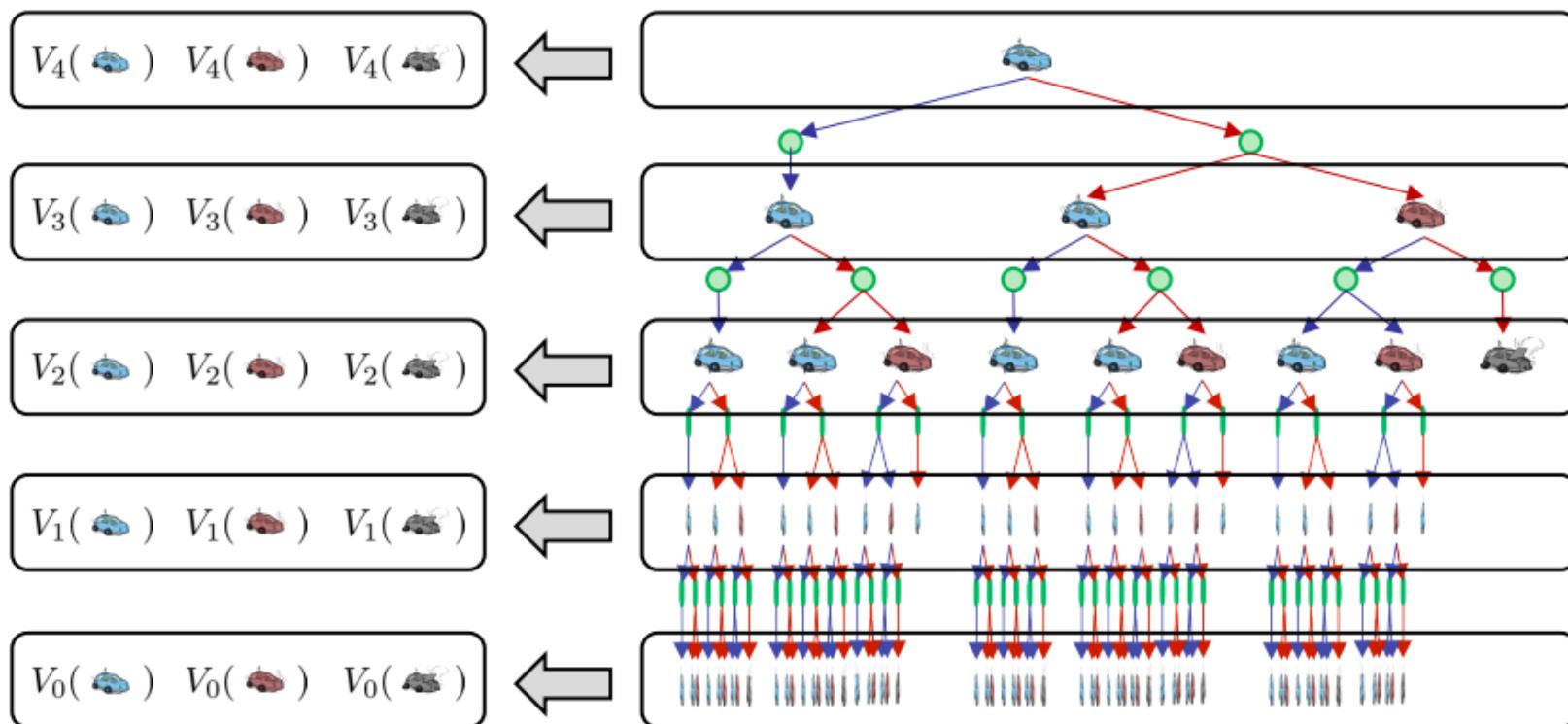


La idea

- Valores para tiempos limitados
- Definir $V_k(s)$ como el valor óptimo de s si el juego termina en k pasos
- De forma equivalente sería la solución de expectimax con un árbol de profundidad k desde s

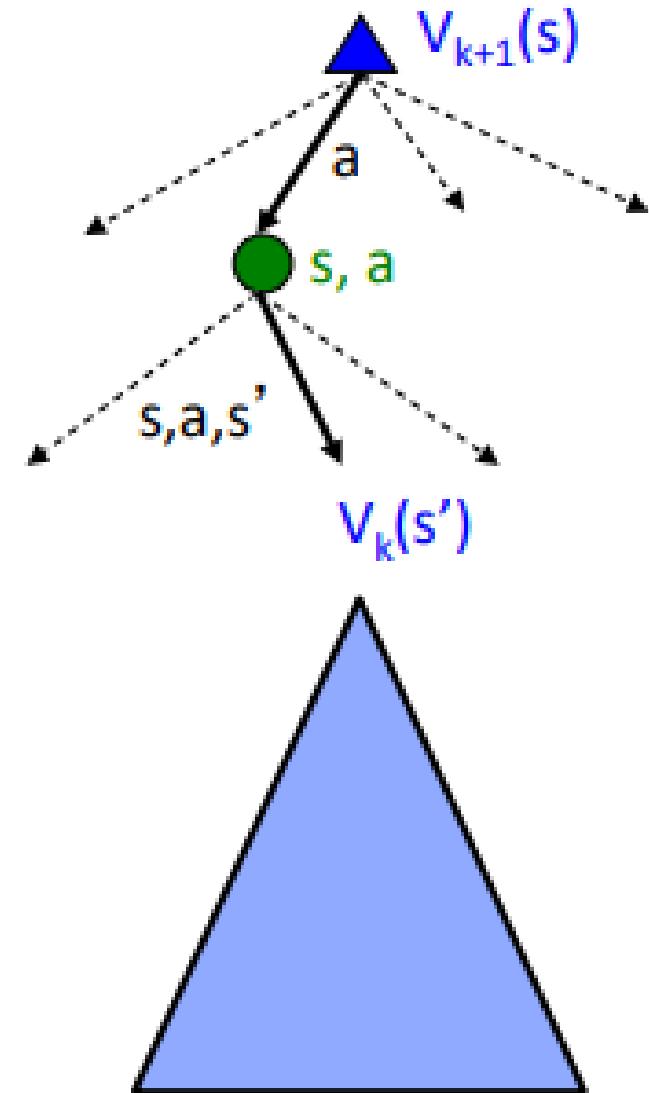


Regresando a nuestro ejemplo



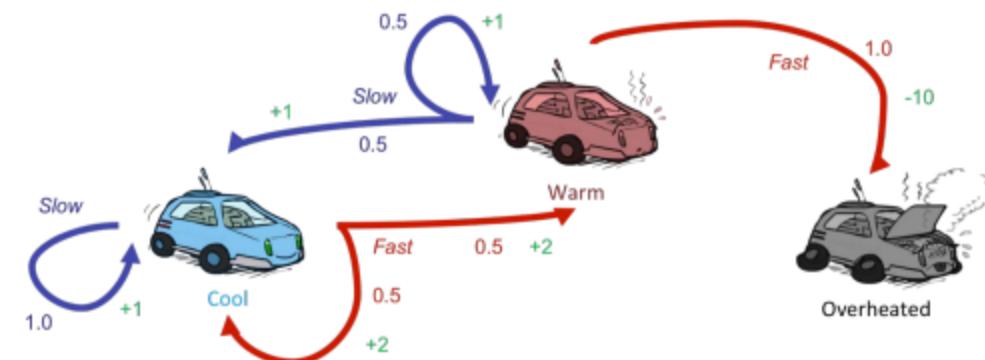
Iteración de valor

- $V_k(s)$ es el valor óptimo de s si el juego termina en k pasos y usar expectimax para solo k pasos
- Iniciamos con $V_0(s) = 0$: ya no hay tiempo y entonces la suma de recompensas esperadas es 0
- Dado el vector de $V_k(s)$, hacer una iteración de expectimax desde cada estado
 - $V_{k+1}(s) = \max_a \sum_{s'} T(s, a, s') [R(s, a, s') + \gamma V_k(s')]$
- Repetir hasta convergencia



Regresemos a nuestro ejemplo

V_2	3.5	2.5	0
V_1	2	1	0
V_0	0	0	0



Assume no discount!

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} T(s, a, s') [R(s, a, s') + \gamma V_k(s')]$$

¿Y si no tenemos T o R?

- Construir un modelo
 - Aprender un modelo aproximado basado en experiencias
 - Resolver los valores como si el modelo fuera el correcto
- Paso 1: aprender el modelo
 - Contar las salidas s' para cada s, a
 - Normalizar para dar un estimado $\hat{T}(s, a, s')$
 - Descubrir cada $\hat{R}(s, a, s')$
- Paso 2: Resolver el proceso de decisión de Markov
 - Utilizando el algoritmo visto previamente

Para la otra vez...

- Aprendizaje por refuerzo II: iteración de política

The End.

