Algoritmos de Toma de Decisiones
Como entretener a tus primos teoría de la decisión

Fada Santiago

FAMAF Facultad de Matemática, Astronomía, Física y Computación

UNC Universidad Nacional de Córdoba

Julio 2025

Contenidos

¿4 en Línea?

"4 en línea", es un juego de mesa para dos jugadores donde el objetivo es ser el primero en alinear cuatro fichas del mismo color, ya sea horizontal, vertical o diagonalmente, en un tablero de 6 x 7 casillas.
Los jugadores alternan turnos colocando una ficha en la parte superior de una columna, y la ficha cae hasta la posición más baja disponible dentro de esa columna

Una perspectiva Matemática

- Se trata de un juego determinista, de suma cero, de información completa y por turnos alternados.
- $\mathcal{A}$, el conjunto de acciones para un jugador, es finito.
- $\mathcal{S}$, el espacio de estados también es finito, aunque grande.
- Se sabe que es un juego **resuelto**: el primer jugador puede forzar la victoria si juega perfectamente.

¿Qué es el Aprendizaje por Refuerzo?

- Un agente toma decisiones en un entorno con el objetivo de maximizar una **recompensa acumulada** a largo del tiempo.
- Cada decisión afecta el estado futuro: se trata de un problema **secuencial** de toma de decisiones.
- El entorno se modela como un **proceso de decisión de Markov** (MDP).

Conceptos clave

agente, **estado**, **acción**, **recompensa**.

Q-Learning

- Algoritmo off-policy que aprende el valor de cada acción en cada estado.
- Aprende una función de valor $Q(s, a)$ que estima la utilidad esperada de tomar una acción a en un estado s.
- Fórmula de actualización:

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left[r + \gamma \max_{a' \in \mathcal{A}} Q(s', a') - Q(s, a) \right]$$

- Se basa en la **diferencia temporal** (TD).
- α es la **tasa de aprendizaje**: cuánto se actualiza el valor anterior. γ es el **factor de descuento**: cuánto pesan las recompensas futuras.

Exploración y Explotación

- Dilema clásico: balancear entre tomar la mejor acción conocida (explotación) y probar nuevas (exploración).
- Estrategias comunes:
  - ϵ-**greedy**: con probabilidad ϵ elige una acción aleatoria.
  - **Softmax**: asigna una probabilidad a cada acción según sus valores $Q(s, a)$ y una **temperatura** τ.
    - τ controla la aleatoriedad: valores altos $\Rightarrow$ distribución más uniforme (más exploración), valores bajos $\Rightarrow$ favorece acciones con mayor Q (más explotación).
- En cualquier caso, tanto ϵ como τ, se suele priorizar la explotación y reducir la exploración con el tiempo.

Implementación en 4 en Línea

- Estados representados como una cadena: tablero + jugador.
- La Q-table almacena valores de $Q(s, a)$ para cada par estado-acción.
- Se usa simetría para mejorar la generalización (espejado horizontal).
- Estrategias de exploración (ϵ-greedy, softmax) así como demás hiperparámetros del MDP totalmente configurables.

Ventajas y Desventajas del Enfoque RL

Parámetros utilizados

- $\alpha = 0.2$.
- $\gamma = 0.9$.
- $\epsilon = 1$, decaimiento $= 0.9995$, $\epsilon_{min} = 0.01$
- estrategia: **softmax**

Ventajas

- Aprende desde cero (sin conocimiento del juego) *.
- Se adapta a oponentes cambiantes.
- Puede generalizar si se entrena correctamente.

Desventajas

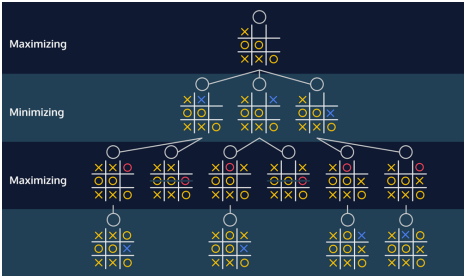
- Requiere mucho entrenamiento *.
- Puede converger a una estrategia subóptima.
- Poca interpretabilidad.

Problemas con este enfoque

- **Alto costo computacional**: requiere millones de partidas de entrenamiento para aprender una política razonable.
- **Aprendizaje ineficiente**: al no tener conocimiento previo del juego, el agente explora muchas jugadas irrelevantes o perdedoras.
- **Convergencia lenta**: depende fuertemente de la elección de hiperparámetros y de una buena política de exploración.
- **Escalabilidad limitada**: al usar una tabla $Q(s, a)$, el espacio de memoria crece rápidamente con la complejidad del estado.
- **Competitividad**: El agente requiere de alguien contra quien jugar.

Minimax clásico

- Recorre el árbol de juego hasta profundidad d.
- Nodos **MAX** (mi turno) $\rightarrow$ busco maximizar la recompensa.
- Nodos **MIN** (rival) $\rightarrow$ minimizar la recompensa.
- Garantiza decisión óptima si d llega al final del juego.
- Puede combinarse con heurísticas y poda alfa-beta para mejorar eficiencia.



Optimización: Poda Alpha-Beta

- Minimax puede volverse muy costoso si el árbol es profundo o ramificado.
- La **poda alpha-beta** evita explorar ramas que no pueden afectar la decisión final.
- Mantiene dos valores:
  - α: mejor valor garantizado para MAX hasta el momento.
  - β: mejor valor garantizado para MIN hasta el momento.
- Si en algún punto se cumple $\alpha \geq \beta$, la rama actual se poda (no se explora).
- Produce la **misma jugada óptima** que Minimax, pero con mucho **menos cómputo**.

Heurística de Evaluación

Idea general

Evalúa el estado del tablero sin llegar al final del juego, estimando quién está en mejor posición.

$$\text{score} = 10^6 \cdot (\text{4 en línea propias}) + 100 \cdot (\text{3+1 propias}) - 120 \cdot (\text{3 rivales}) + \dots$$

- Analiza todas las "ventanas" posibles de 4 celdas.
- Premia configuraciones ofensivas propias (alineaciones de 2, 3, 4).
- Penaliza amenazas del oponente (3 en línea con hueco).
- Otorga mayor valor a posiciones centrales (más estratégicas).
- Se utiliza en nodos intermedios del árbol de Minimax para cortar la búsqueda.

Planificación vs. Aprendizaje

	Minimax	**Q-Learning**
Entrenamiento	–	Alto
Almacenamiento	–	Muy Alto, almaceno toda la tabla
Online cost	Alto (si d grande)	Bajo (tabla)
Optimalidad	Sí (si d grande)	Aproximada
Generaliza	No	Sí (con experiencia)
Explicabilidad	Alta	Media

Demo

Jugadores

- Minimax.
- Q-Learning.
- Humano.

Conclusiones y Preguntas

- **Minimax** sigue una estrategia de planificación determinista, útil cuando se conoce completamente el entorno y se prioriza la exactitud.
- **Q-Learning** aprende por interacción, sin conocer el modelo, y permite adaptarse a entornos dinámicos.
- La elección depende de los objetivos: precisión vs. adaptabilidad, conocimiento del entorno vs. experiencia.
- Ambos enfoques conectan con temas centrales del curso, asi como temas relacionados:
  - Juegos secuenciales y multiagente.
  - Búsqueda heurística y optimización.
  - Aprendizaje por refuerzo y TD-learning.
- Futuras extensiones: Deep Q-learning o agentes mixtos.

Muchas gracias!
Preguntas?