

2025

# Informe Obligatorio 2:

Juegos alternados de información perfecta e imperfecta

---



Facultad de Ingeniería - Bernard-Wand Polak

Curso: Sistemas Multiagentes

Valeria Eskenazi – 326684

3 de Julio 2025

	.....	0
<b>1</b>	<b>INTRODUCCIÓN .....</b>	<b>3</b>
1.1	Objetivo.....	3
1.2	Validación .....	3
<b>2</b>	<b>IMPLEMENTACIÓN .....</b>	<b>4</b>
2.1	Estructura general del Código.....	4
2.2	Algoritmos implementados .....	5
<b>3</b>	<b>AMBIENTES UTILIZADOS .....</b>	<b>6</b>
3.1	TicTacToe .....	7
3.2	Nocca Nocca.....	7
3.2.1	Funcion de evaluación .....	8
3.2.1.1	Verificacion si ordena los estados.....	8
3.2.1.2	Facil de evaluar.....	9
3.2.1.3	Estar relacionado con ganar .....	9
3.3	Khun poker (2 y 3 jugadores).....	10
<b>4</b>	<b>VALIDACIÓN Y RESULTADOS EXPERIMENTALES .....</b>	<b>11</b>
4.1	Juegos alternados de información perfecta.....	12
4.1.1	TicTacToe .....	12
4.1.1.1	MinMax vs MinMAX .....	12
4.1.1.2	MinMax vs Random.....	15
4.1.1.3	MCTS vs MCTS .....	17
4.1.1.4	MCTS vs RandomAgent.....	20
4.1.1.5	MCTS vs MinMax .....	23
4.1.1.6	Resumen .....	26
4.1.2	Nocca Nocca .....	28
4.1.2.1	MinMax depth=1 vs Random .....	28
4.1.2.2	MinMax Depth=1 vs MinMax Depth=2 .....	32
4.1.2.3	MCTS simulations=4 rollouts=2 vs Random .....	34
4.1.2.4	MCTS vs MCTS .....	38
4.1.2.5	MCTS vs MinMax .....	40
4.1.2.6	Resumen .....	42
4.2	Juegos alternados de información imperfecta .....	44
4.2.1	Khun poker 2 jugadores .....	44
4.2.1.1	Entrenamiento corto (10 iteraciones) y Corrida de estudio de 2000 partidas: .....	44
4.2.1.2	Entrenamiento corto (10 iteraciones) y Corrida de estudio de 10.000 partidas ....	49
4.2.1.3	Entrenamiento medio (1000 iteraciones) y Corrida de estudio de 10.000 partidas	53
4.2.1.4	Entrenamiento largo (100.000 iteraciones) y Corrida de estudio de 10.000 partidas	57

4.2.1.5	Resumen .....	61
4.2.2	<b>Khun poker 3 jugadores .....</b>	62
4.2.2.1	Entrenamiento corto (10 iteraciones) y Corrida de estudio de 1.000 partidas .....	62
4.2.2.2	Entrenamiento corto (10 iteraciones) y Corrida de estudio de 10.000 partidas ....	66
4.2.2.3	Entrenamiento media (1.000 iteraciones) y Corrida de estudio de 1.000 partidas	70
4.2.2.4	Entrenamiento medio (1.000 iteraciones) y Corrida de estudio de 10.000 partidas	74
4.2.2.5	Entrenamiento largo (100.000 iteraciones) y Corrida de estudio de 1.000 partidas	78
4.2.2.6	Entrenamiento largo (100.000 iteraciones) y Corrida de estudio de 10.000 partidas	82
4.2.2.7	Resumen .....	86
<b>5</b>	<b>CONCLUSIONES GENERALES.....</b>	<b>87</b>
5.1	<b>Juegos alternados de información perfecta.....</b>	<b>87</b>
5.2	<b>Juegos alternados de información perfecta.....</b>	<b>89</b>
<b>6</b>	<b>ANEXO .....</b>	<b>91</b>

# 1 Introducción

El presente informe documenta el desarrollo y análisis del trabajo práctico centrado en el estudio de algoritmos de aprendizaje en entornos multiagente del tipo alternados.

## 1.1 Objetivo

El objetivo principal de este trabajo es implementar y validar diferentes estrategias de decisión y adaptación para agentes autónomos que interactúan en escenarios de juegos alternados de información perfecta e imperfecta. Se buscó analizar el desempeño y las características de algoritmos representativos del aprendizaje multiagente, abordando distintos enfoques para la toma de decisiones en entornos competitivos.

Para ello, se implementaron los siguientes algoritmos:

- **MonteCarlo Tree Search (MCTS)**
- **MinMAX (MM),**
- **Counterfactual regret minimization (CRM)**

Cada uno de estos algoritmos representa una perspectiva distinta frente al problema del aprendizaje multiagente, variando en su modo de exploración, actualización de políticas y adaptación según la estructura y el tipo de información disponible en el entorno.

## 1.2 Validación

La validación se realizó mediante enfrentamientos entre agentes, utilizando una serie de ambientes provisto por la cátedra:

- **TicTacToe**
- **Nocca Nocca**
- **Khun Poker (2 jugadores)**
- **Khun Poker (2 jugadores)**

En los casos de TicTacToe y Nocca Nocca, la comparación entre agentes se basó en el análisis de la tasa de victoria (win rate) y el tiempo de decisión de cada algoritmo.

Para Kuhn Poker (2 y 3 jugadores), además de la tasa de victoria, se analizaron métricas adicionales como la evolución de la política, la frecuencia de las acciones seleccionadas, la recompensa acumulada y la distancia al equilibrio de Nash, con el objetivo de capturar el aprendizaje estratégico y la adaptación en un entorno de información imperfecta.

## 2 Implementación

### 2.1 Estructura general del Código

La implementación del trabajo se organizó en módulos separados que facilitan la escalabilidad, la reutilización del código y la comparación entre algoritmos. A continuación se describe la estructura general:

- **base/**: contiene las clases abstractas comunes:
  - game.py
  - agent.py
  - utils.py
- **agents/**: contiene los agentes con sus respectivos algoritmos de aprendizaje:
  - minmax.py: implementación de MinMAX.
  - mcts\_t.py: implementación de MonteCarlo Tree Search.
  - counterfactualregret.py: implementación Counter Factual Regret
  - Agent\_random.py: implementacion agente randomico
- **games/**: cada carpeta contiene la implementación de los entornos de juegos utilizados:
  - tictactoe/

- nocca\_nocca/
  - khun\_poker/
- **Obligatorio:** Contine notebooks de implementacion asi como su correspondiente script auxiliar. Organizado de la siguiente manera:
  - **TicTacToe:**
    - tictactoe.ipynb
    - torneo\_tictactoe.py
  - **Nocca\_Nocca:**
    - Nocca\_Nocca.ipynb
    - torneo\_nocca\_nocca.py
  - **Khun\_poker:**
    - Khun\_2.ipynb
    - Khun\_3.ipynb
    - Torneo\_khun\_poker.py

## 2.2 Algoritmos implementados

A continuación, se detallan los algoritmos implementados, todos ellos heredan de la clase

Agent y están integrados al marco común de juego simultáneo:

- **Monte Carlo Tree Search (MCTS):** El agente se desarrolla siguiendo la estructura clásica del algoritmo MCTS. Se utiliza un template base sobre el cual se construyen las etapas fundamentales: selección, expansión, simulación y backpropagación. La implementación respeta la lógica estándar del método, adaptando la estructura de árbol de acuerdo al entorno evaluado.
- **Nocca Nocca:** Para este entorno se diseña una función de evaluación específica, que combina linealmente dos factores principales: la distancia a la meta y la cantidad de fichas del oponente que el agente logra bloquear. Se experimenta con dos definiciones de distancia: la distancia vertical (camino

más corto en dirección a la meta) y la distancia Manhattan (considerando desplazamientos horizontales y verticales). Ambas alternativas son evaluadas comparativamente en la notebook.

- **Counterfactual Regret Minimization (CFR):** El agente CFR se implementa completando el template base provisto, siguiendo la lógica vista en clase para la actualización de políticas y la acumulación de regrets contrafactuals. Esta estructura permite al agente adaptarse a escenarios de información imperfecta como Kuhn Poker.
- **MinMax:** El agente MinMax se basa en el código provisto por la cátedra, implementando la búsqueda exhaustiva de las mejores jugadas considerando la profundidad completa del árbol de juego, adecuado para entornos de información perfecta.
- **Random Agent:** Se incluye también un agente aleatorio, utilizando el código base suministrado por la cátedra, como referencia de comportamiento no estratégico en los distintos entornos.

### 3 Ambientes utilizados

A lo largo del trabajo se utilizaron distintos entornos de **juegos alternados**, diseñados para evaluar el comportamiento de los agentes bajo diversas condiciones y estructuras de turno. Cada ambiente presenta dinámicas particulares —desde información perfecta hasta información imperfecta— que permiten analizar diferentes aspectos de los algoritmos implementados, tales como la capacidad de planificación secuencial, la adaptación frente a estrategias cambiantes de los oponentes y la eficiencia en la toma de decisiones por turnos.

Estos entornos permiten estudiar propiedades relevantes como la convergencia hacia estrategias óptimas, la robustez frente a distintos tipos de adversarios y el desempeño bajo restricciones de tiempo o información.

### **3.1 TicTacToe**

Juego clásico de información perfecta y suma cero, donde dos agentes alternan movimientos en un tablero 3x3, buscando formar una línea de tres símbolos iguales. Este entorno permite evaluar la eficiencia y robustez de los algoritmos en un contexto determinístico y totalmente observable.

En este ambiente se utilizaron principalmente los algoritmos MinMax y MCTS. Para MinMax, se estudiaron diferentes profundidades del árbol y su impacto tanto en el índice de ganancia como en los tiempos de ejecución. En el caso de MCTS, se exploraron distintas configuraciones de parámetros, tales como la cantidad de simulaciones y la estrategia de rollout, evaluando su influencia sobre el desempeño del agente.

Las configuraciones utilizadas en este ambiente fueron

- MinMax vs MinMax (con diferentes profundidades)
- MinMax vs Random
- MCTS vs MCTS (con diferentes simulaciones y rollout)
- MCTS vs Random
- MCTS vs MinMax (utilizando los mejores para cada caso)

### **3.2 Nocca Nocca**

Nocca Nocca es un juego de tablero por turnos que introduce una dinámica más compleja de movimiento y bloqueo de fichas. La estructura del juego requiere que los agentes planifiquen secuencias de acciones considerando tanto su avance hacia la meta como la posibilidad de obstruir al oponente.

En este entorno, se fue preciso utilizar las configuraciones de los agentes más sencillas que las desarrollados para TicTacToe, ya que al tratarse de un juego más complejo, aumentar la profundidad/complejidad de la toma de decisiones, genera un aumento exponencial en el tiempo y recursos computacionales, no siendo la idea de esta notebook. Trabajar con agentes sencillos, permite centrar el análisis en la función de evaluación específica del agente.

### 3.2.1 Función de evaluación

Para evaluar el desempeño de los agentes en este juego, se propuso la siguiente función de evaluación:

$$E(s) = \alpha_B * B(s) - \alpha_D * D(s) + \text{Bono_Terminal}$$

Con:

- $D(s)$  representa la diferencia de distancia hacia la meta entre ambos jugadores.
- $B(s)$  contabiliza la diferencia de fichas bloqueadas a favor del agente.
- $\alpha_B$  y  $\alpha_D$  son los pesos asignados a cada componente.

Definición de términos:

$$D(s) = Da(s) - Do(s) \text{ con:}$$

- $Da(s)$  distancia del agente a la casa del oponente
- $Do(s)$  distancia del oponente a la casa del agente

$$B(s) = Ba(s) - Bo(s) \text{ con:}$$

- $Ba(s)$  cantidad de fichas del oponente bloqueadas por el agente
- $Bo(s)$  cantidad de fichas del agente bloqueadas por el oponente

#### 3.2.1.1 Verificación si ordena los estados

Si  $s=win$ :

- Caso 1:  $s=win$  porque agente bloqueó todas las fichas:
  - $Ba(s)=5$  y por el contra parte,  $Bo(s)$  tiene que ser  $<5$ , entonces  $0 < B(s) < 5$
  - $Da(s)$  puede ser igual o distinto que  $Do(s)$ , por lo que  **$D(s)$  puede ser negativo o positivo**
- Caso 2:  $s=win$  porque el agente alcanzó la casa del oponente:
  - $Ba(s)$  puede ser igual o distinto que  $Bo(s)$ , por lo que  **$B(s)$  puede ser negativo o positivo**.
  - $Da(s)=0$ , y  **$D(s)<0$** .

Si  $s=loss$ :

- Caso 1:  $s=loss$  porque oponente bloqueó todas las fichas:
  - $Bo(s)=5$  y por el contra parte,  $Ba(s)$  tiene que ser  $<5$ , entonces  $B(s)<0$
  - $Da(s)$  puede ser igual o distinto que  $Do(s)$ , por lo que  **$D(s)$  puede ser negativo o positivo**
- Caso 2:  $s=loss$  porque el agente alcanzó la casa del oponente:
  - $Ba(s)$  puede ser igual o distinto que  $Bo(s)$ , por lo que  **$B(s)$  puede ser negativo o positivo**.
  - $Do(s)=0$ , y  **$D(s)>0$** .

Debido a la existencia de dos condiciones terminales (bloqueo y llegada a la meta), los pesos de la función de evaluación deben adaptarse al contexto. Para asegurar un correcto ordenamiento de estados, se propone:

$$\alpha_B(s) = \begin{cases} 0 & \text{si } D_a(s) \text{ o } D_o(s) = 0 \\ \alpha_B & \text{en otro caso} \end{cases}$$

$$\alpha_D(s) = \begin{cases} 0 & \text{si } B_a(s) \text{ o } B_o(s) = 5 \\ \alpha_D & \text{en otro caso} \end{cases}$$

De esta manera:

- Si la victoria (o derrota) es por bloqueo total, solo se tiene en cuenta  $B(s)$ .
- Si la victoria (o derrota) es por llegar a la meta, solo se tiene en cuenta  $D(s)$ .

### *3.2.1.2 Facil de evaluar*

Se consideraron dos alternativas para la función de distancia: vertical y Manhattan. Ambas comparten la lógica de identificar las fichas disponibles (no bloqueadas), pero difieren en cómo calculan la distancia efectiva a la meta. La versión Manhattan incorpora un costo adicional para movimientos laterales, simplificando el cálculo en comparación con una búsqueda exacta de caminos posibles, lo que reduce la complejidad computacional. La función de bloqueo se implementó mediante dos bucles anidados que contabilizan las fichas efectivamente bloqueadas.

En ambos casos, la eficiencia computacional queda reflejada en los tiempos de ejecución de los agentes, lo que permite analizar la practicidad de cada alternativa.

### *3.2.1.3 Estar relacionado con ganar*

La función de evaluación está directamente alineada con las condiciones de victoria del juego. El uso del bono terminal y la asignación dinámica de pesos aseguran que los estados finales sean correctamente priorizados, mientras que en los estados intermedios, la función guía a los agentes a maximizar sus probabilidades de ganar ya sea bloqueando al oponente o llegando primero a la meta.

### **3.3 Khun poker (2 y 3 jugadores)**

Versión simplificada del póker, que representa un entorno de información imperfecta y azar, donde los jugadores alternan acciones de apuesta, pase o igualar. Este ambiente es especialmente útil para analizar la capacidad de los agentes de adaptarse estratégicamente en situaciones de incertidumbre, modelando el comportamiento de los oponentes y evaluando la eficacia de distintas estrategias en un contexto no determinístico.

En este entorno, el análisis incluyó métricas adicionales más allá del índice de victoria, como la evolución de la política de juego, la frecuencia de las acciones seleccionadas, la recompensa acumulada y la distancia al equilibrio de Nash. Estas métricas permitieron evaluar la capacidad de aprendizaje, adaptación y aproximación a estrategias óptimas de los algoritmos implementados

En este caso, se trabajó únicamente con el agente CFR.

## 4 Validación y resultados experimentales

Con el objetivo de evaluar el comportamiento y desempeño de los algoritmos implementados, se diseñó una serie de experimentos controlados para cada uno de los entornos previamente descritos. Para ello, se desarrollaron notebooks específicas para cada juego, en las cuales se implementó una script específico para cada entorno con el objetivo de disputar partidas individuales entre dos agentes y, posteriormente, organizar torneos consistentes en la ejecución de N partidas entre diferentes configuraciones de agentes.

El enfoque experimental y las métricas analizadas varían según el entorno:

- **TicTacToe:** Se realizaron torneos entre diferentes agentes, promediando resultados como la tasa de victoria y los tiempos de ejecución. Además, se analizó la evolución de la tasa de ganancia a lo largo de las partidas para observar tendencias y estabilidad en el desempeño.
- **Nocca Nocca:** Al igual que en TicTacToe, se midieron la tasa de victoria y los tiempos de ejecución. Adicionalmente, se exploró cómo varían estos indicadores al modificar la función de evaluación empleada por los agentes, permitiendo comparar distintas configuraciones y analizar su impacto en el rendimiento.

Con este fin se desarrolló el script de evaluación se estructuró en dos niveles:

- El primer nivel consiste en analizar la evolución de los agentes a lo largo de un torneo, utilizando una configuración específica de la función de evaluación.
- El segundo nivel corresponde a la comparación entre los diferentes torneos, permitiendo contrastar el desempeño de los agentes según las distintas funciones de evaluación utilizadas en cada caso.
- **Kuhn Poker:** El análisis se centró en métricas como la probabilidad de recibir y jugar cada carta, la probabilidad de realizar apuestas, el regret acumulado y la distancia al equilibrio de Nash, para un agente cfr.

## 4.1 Juegos alternados de información perfecta

### 4.1.1 TicTacToe

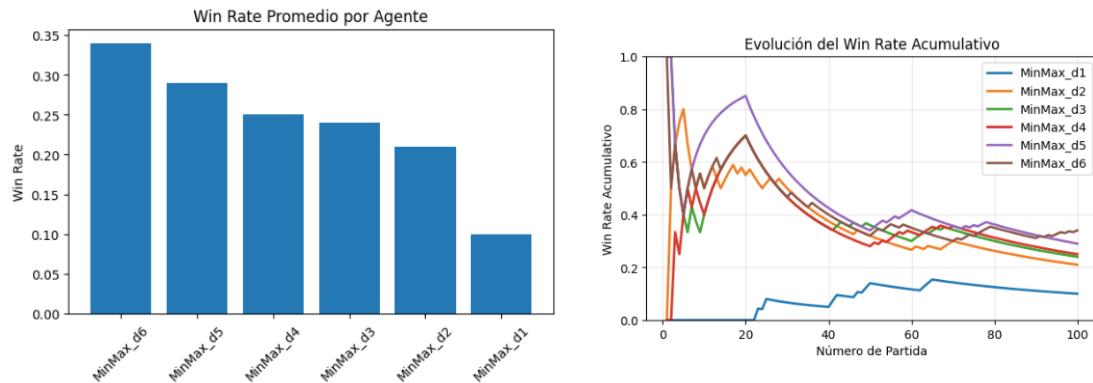
Se puede encontrar la notebook dentro de Juegos\_Alternados / Obligatorio / TicTacToe/ TicTacToe\_torne.ipynb

#### 4.1.1.1 MinMax vs MinMax

Las configuraciones probadas para este agente se detallan a continuación:

- MniMax\_1: depth=1
- MniMax \_2: depth=2
- MniMax \_3: depth=3
- MniMax \_4: depth=4
- MniMax \_5: depth=5
- MniMax \_6: depth=6

#### Evolución del win rate:

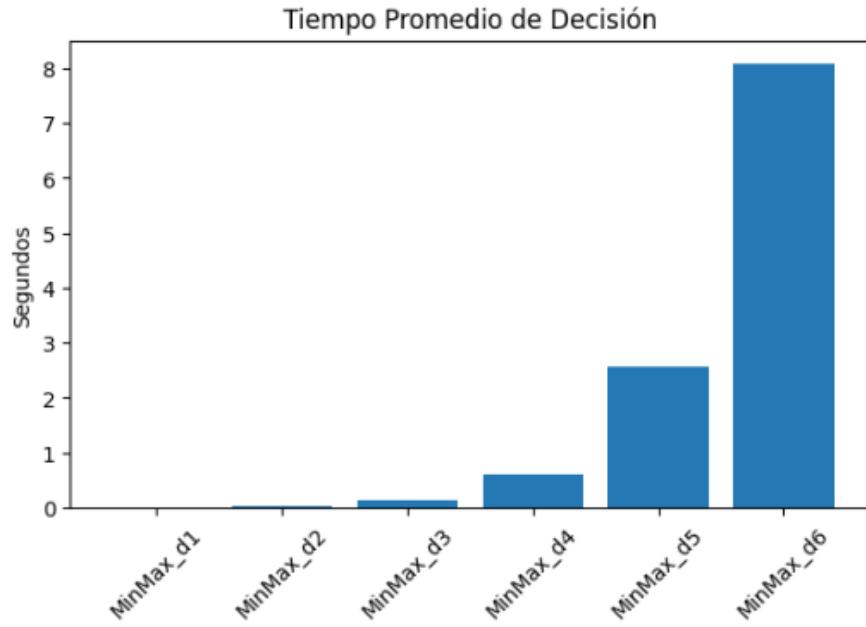


El gráfico de barras muestra el win rate promedio alcanzado por cada configuración de profundidad. Se observa que a medida que aumenta la profundidad del árbol (de d1 a d6), el win rate promedio también aumenta. Esto indica que agentes MinMax con mayor profundidad exploran más posibles jugadas y toman decisiones más óptimas, lo que se traduce en un mejor desempeño general.

El gráfico de líneas presenta cómo evoluciona el win rate acumulado a lo largo de 100 partidas para cada configuración. Las profundidades mayores tienden a

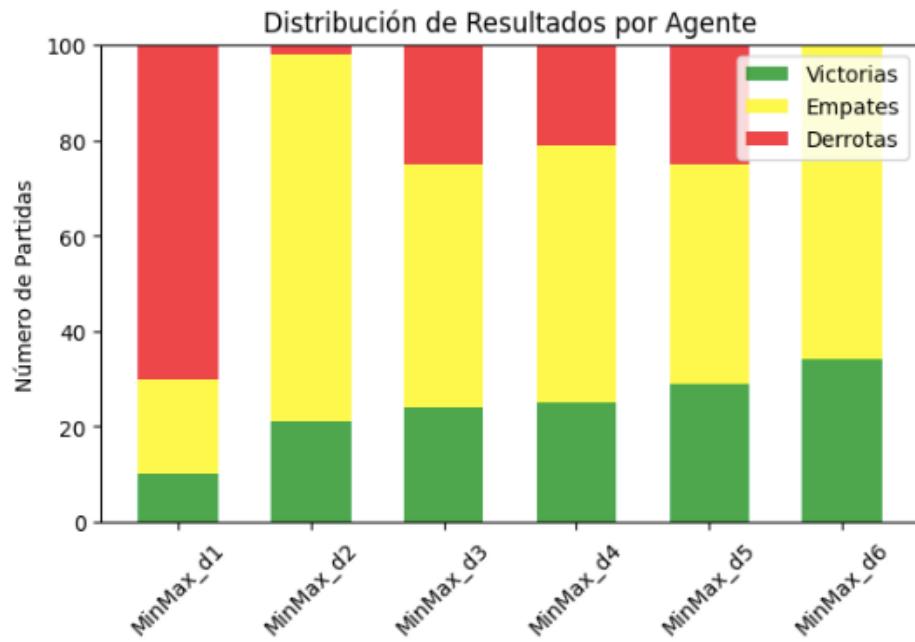
estabilizarse en valores más altos de win rate, mientras que los agentes de menor profundidad mantienen desempeños más bajos de manera consistente.

Tiempo promedio de decisión:



Se observa un crecimiento exponencial en el tiempo promedio de decisión por jugada a medida que aumenta la profundidad del árbol. Para las profundidades mayores (d5 y d6), el tiempo por movimiento puede superar los 2 y 8 segundos respectivamente, lo que representa un costo computacional considerable para obtener un mejor desempeño.

## Distribución de Resultados por Agente



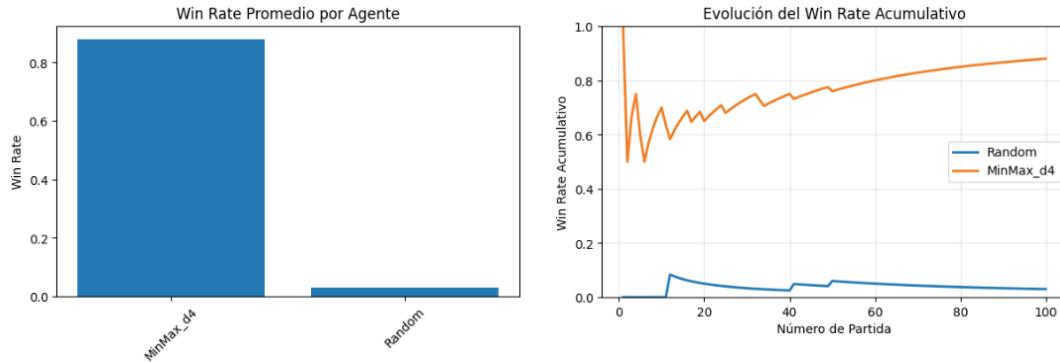
El gráfico de barras apiladas muestra la proporción de victorias, empates y derrotas para cada configuración. Los agentes con baja profundidad tienen una alta cantidad de derrotas y pocos empates/victorias. A medida que la profundidad aumenta, disminuyen las derrotas, aumentan los empates y suben las victorias, mostrando que MinMax es capaz de encontrar mejores estrategias a medida que se incrementa la profundidad de búsqueda.

## Conclusiones de los enfrentamientos:

Aumentar la profundidad del agente MinMax en TicTacToe mejora de manera significativa su win rate y su robustez ante el oponente, pero implica un incremento considerable en el tiempo de cómputo por jugada. En este caso, la profundidad 4 se seleccionó como la más adecuada, ya que ofrece el mejor trade-off entre desempeño (win rate alto y reducción de derrotas) y tiempos de decisión razonables para aplicaciones prácticas.

#### 4.1.1.2 MinMax vs Random

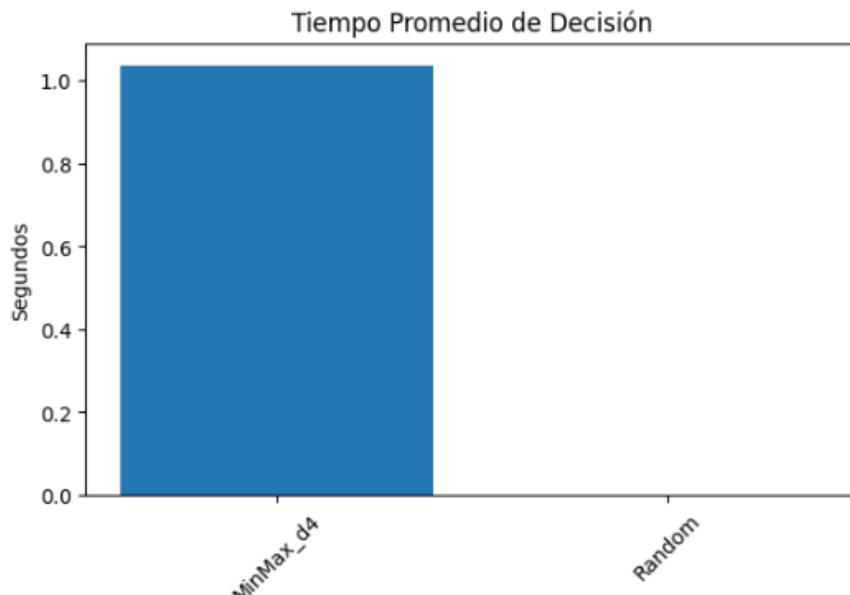
##### Evolución del win rate:



El agente MinMax\_d4 alcanza un win rate promedio superior al 85%, mientras que el agente Random apenas supera el 3%. Esto demuestra que MinMax con profundidad adecuada es capaz de dominar casi por completo a un agente que elige sus acciones al azar.

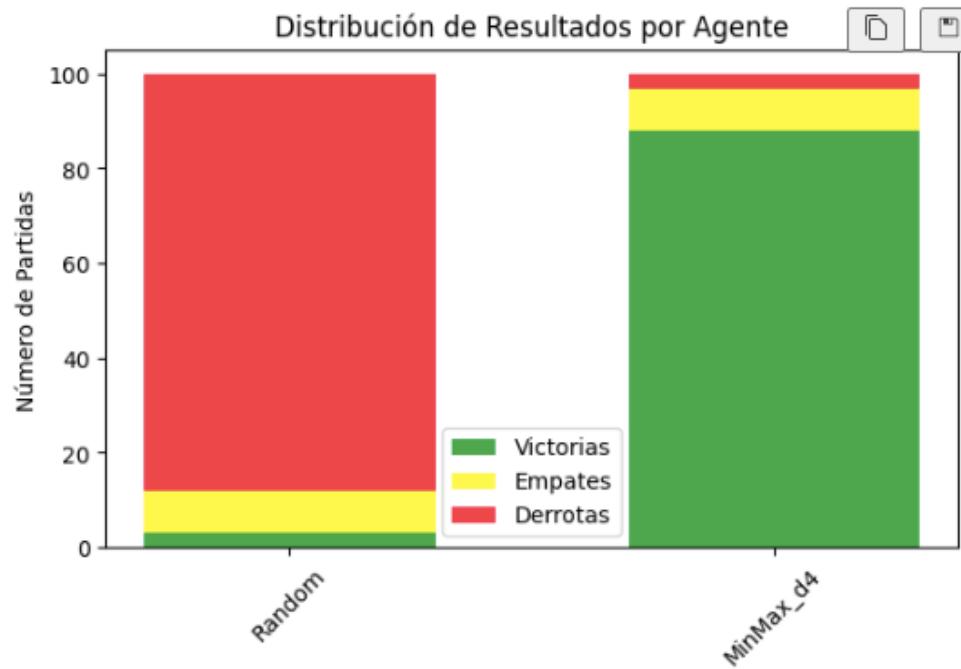
Esto también queda demostrado en el grafico de evolución de win rate acumulativo del torneo. A lo largo de las 100 partidas, MinMax\_d4 mantiene un win rate acumulado alto y estable, mientras que el agente Random permanece en valores muy bajos durante todo el torneo. La diferencia de desempeño es evidente y consistente desde el inicio.

##### Tiempo promedio de decisión:



El tiempo de decisión de MinMax\_d4 es sensiblemente mayor (alrededor de 1 segundo por jugada), mientras que el agente Random resuelve sus movimientos de forma prácticamente instantánea. Esto refleja el costo computacional asociado al uso de algoritmos de búsqueda exhaustiva, en contraste con la simpleza del agente aleatorio.

#### Distribución de Resultados por Agente



El gráfico de barras apiladas muestra que MinMax\_d4 obtuvo la gran mayoría de las victorias, registrando apenas algunas derrotas y empates. Por el contrario, el agente Random fue derrotado en casi todas las partidas, con una proporción mínima de victorias o empates.

#### Conclusiones de los enfrentamientos:

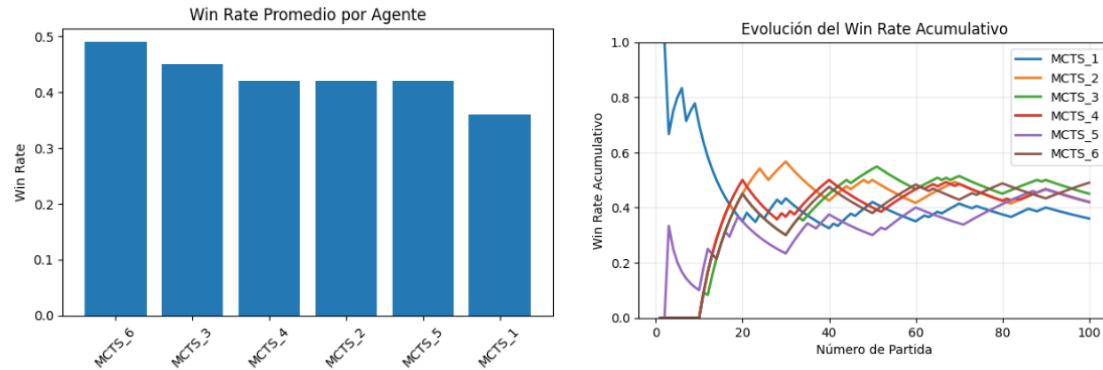
La comparación demuestra la efectividad del agente MinMax con una profundidad razonable en un entorno como TicTacToe, superando de forma contundente a un agente sin estrategia. El costo en tiempo de decisión es mayor, pero se justifica ampliamente por el salto de desempeño frente a alternativas simples.

#### 4.1.1.3 MCTS vs MCTS

Se evaluó el desempeño del agente Monte Carlo Tree Search (MCTS) en TicTacToe, variando los parámetros de cantidad de simulaciones y rollouts. Las configuraciones probadas se detallan a continuación:

- MCTS\_1: 50 simulaciones, 10 rollouts
- MCTS\_2: 50 simulaciones, 20 rollouts
- MCTS\_3: 50 simulaciones, 30 rollouts
- MCTS\_4: 100 simulaciones, 10 rollouts
- MCTS\_5: 100 simulaciones, 20 rollouts
- MCTS\_6: 100 simulaciones, 30 rollouts

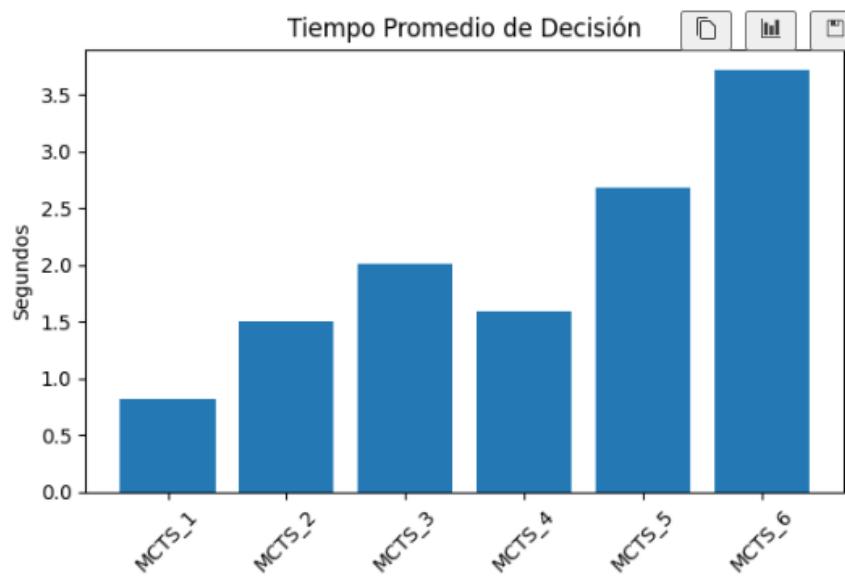
#### Evolución del win rate:



Win Rate Promedio: El win rate promedio varía entre las distintas configuraciones. La configuración MCTS\_5 (100 simulaciones, 20 rollouts) alcanza el mejor desempeño, seguida por MCTS\_4 y MCTS\_3. En general, se observa que aumentar la cantidad de simulaciones tiene un impacto positivo, aunque no estrictamente lineal, y el valor óptimo de rollouts parece estar en torno a 20.

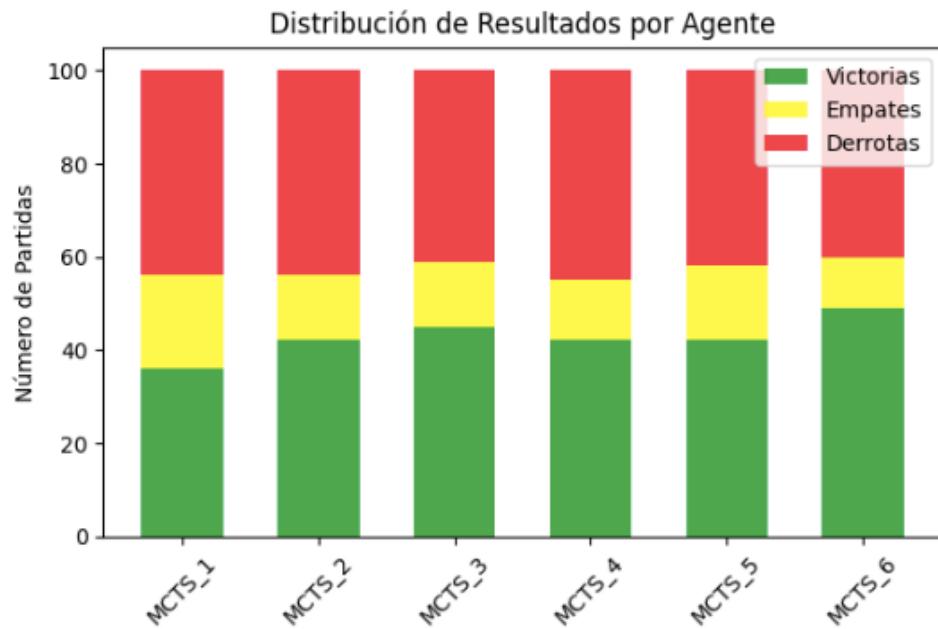
Evolución del Win Rate: El gráfico de la derecha muestra la evolución del win rate acumulativo a lo largo de 100 partidas. Todas las configuraciones tienden a estabilizarse en valores similares tras un período inicial de fluctuación, aunque las mejores configuraciones (especialmente MCTS\_5) logran mantenerse en niveles superiores.

### Tiempo promedio de decisión:



El tiempo necesario para que el agente tome una decisión aumenta conforme se incrementan la cantidad de simulaciones y rollouts. En particular, la configuración MCTS\_6 (100 simulaciones, 30 rollouts) tiene el tiempo de decisión más alto, superando los 3.5 segundos por jugada, mientras que MCTS\_1 (50 simulaciones, 10 rollouts) se mantiene por debajo de 1 segundo. Esto evidencia el clásico trade-off entre mayor capacidad de búsqueda/exploración y eficiencia computacional.

## Distribución de Resultados por Agente



El gráfico de barras apiladas muestra que, si bien existen diferencias entre configuraciones, las proporciones de victorias, empates y derrotas se mantienen relativamente estables en todas las variantes de MCTS evaluadas. No se observa una diferencia significativa en la distribución de resultados al modificar el número de simulaciones y rollouts, lo que sugiere que, dentro del rango probado, los parámetros de MCTS no impactan drásticamente en el balance general de resultados.

## Conclusiones de los enfrentamientos:

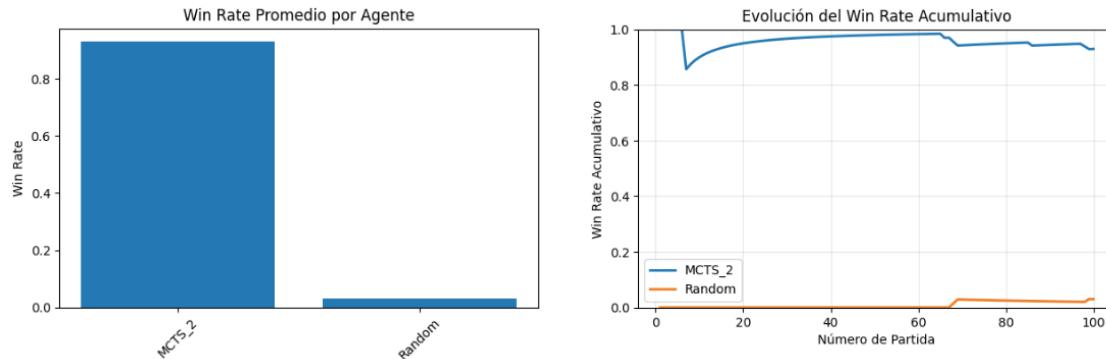
Se observa que el desempeño (win rate) mejora al incrementar el número de simulaciones y rollouts, aunque este efecto no es estrictamente lineal y se ve limitado por el costo computacional.

El mejor trade-off entre rendimiento y eficiencia se alcanza con la configuración MCTS\_2 (50 simulaciones, 20 rollouts). Este agente logra un win rate competitivo, manteniendo el tiempo de decisión por jugada en valores razonablemente bajos (alrededor de 1.5 segundos). Configuraciones más exigentes (como MCTS\_5 o MCTS\_6) ofrecen solo una mejora marginal en el desempeño, pero a costa de un incremento significativo en los tiempos de cómputo.

#### 4.1.1.4 MCTS vs RandomAgent

Utilizando las conclusiones del apartado anterior, se propone el enfrentamiento entre MCTS\_2: 50 simulaciones, 20 rollouts, y Agente randomico

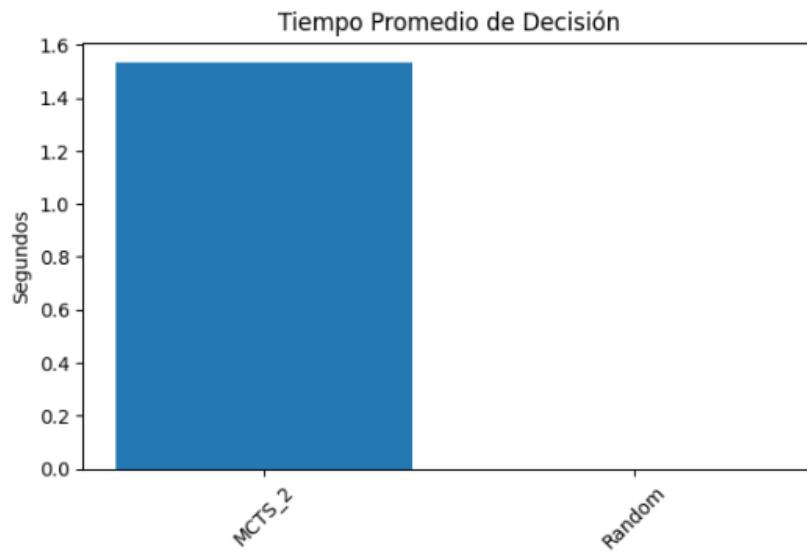
##### Evolución del win rate:



Win Rate Promedio por Agente: El agente MCTS\_2 alcanza un win rate promedio superior al 90%, mientras que el agente Random apenas supera el 3%. Esta diferencia drástica evidencia la enorme ventaja estratégica que aporta el uso de MCTS en comparación con decisiones aleatorias.

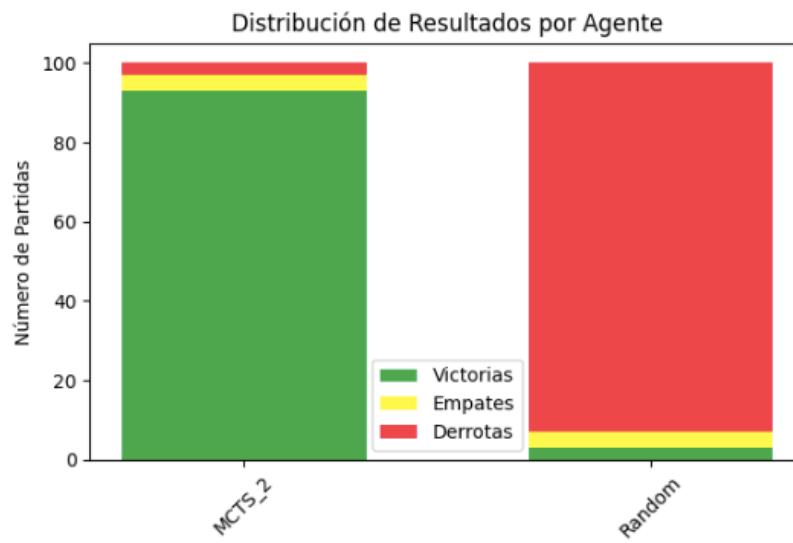
Evolución del Win Rate Acumulativo: A lo largo de las 100 partidas disputadas, MCTS\_2 mantiene un win rate acumulado muy alto y estable, cercano a 0.95, mientras que el agente Random permanece prácticamente sin victorias durante toda la secuencia.

### Tiempo promedio de decisión:



El tiempo de decisión para MCTS\_2 se mantiene en torno a 1.5 segundos por jugada, considerablemente mayor al del agente Random, que responde prácticamente de forma instantánea. Sin embargo, este tiempo es razonable para aplicaciones prácticas considerando el gran salto en desempeño.

### Distribución de Resultados por Agente



La distribución de resultados muestra que MCTS\_2 obtiene la gran mayoría de las victorias, con solo unos pocos empates y derrotas. En cambio, el agente Random

es derrotado en casi todas las partidas, apenas logrando empates y victorias marginales.

Conclusiones de los enfrentamientos:

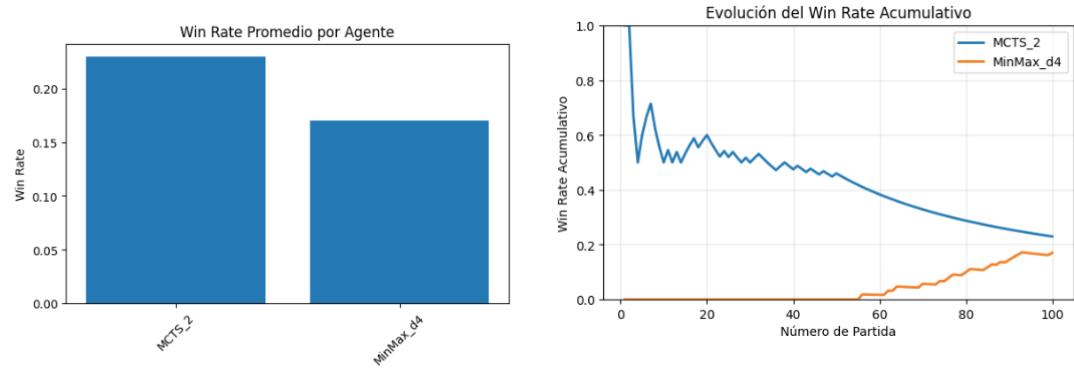
El enfrentamiento entre el agente MCTS\_2 y el agente Random en TicTacToe demuestra la efectividad y robustez de MCTS para este entorno. El salto en el win rate, junto con una baja frecuencia de derrotas y un tiempo de decisión aceptable, confirman que MCTS es claramente superior a estrategias aleatorias, incluso con parámetros moderados de simulación y rollout. Esto valida el uso de MCTS como una estrategia sólida para juegos determinísticos y de información perfecta como TicTacToe.

#### 4.1.1.5 MCTS vs MinMax

De los casos previsos, se enfrenta los agentes con mejores mejores trade off:

- MCTS\_2: 50 simulaciones, 20 rollouts
- MinMax: depth=4

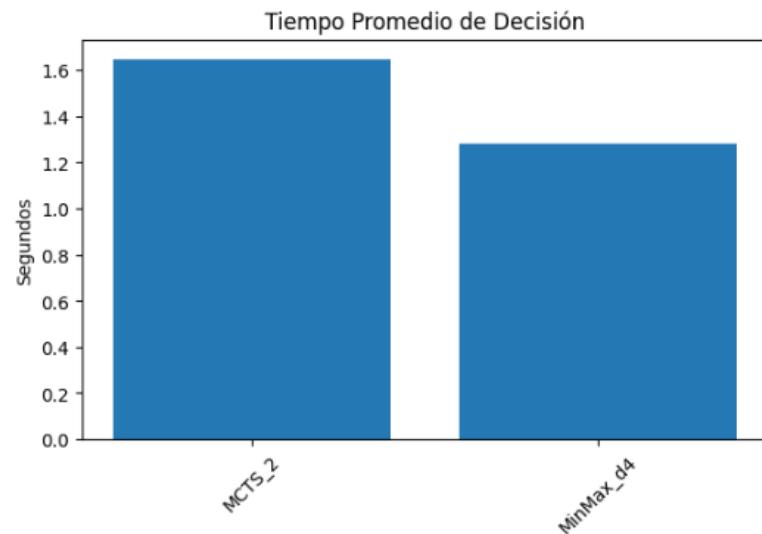
Evolución del win rate:



Win Rate Promedio por Agente: El agente MCTS\_2 obtiene un win rate promedio mayor que MinMax\_d4 (alrededor de 0.23 frente a 0.17). Aunque ambos agentes logran superar al azar, ninguno domina completamente al otro, lo que sugiere un enfrentamiento relativamente parejo, aunque con ventaja para MCTS.

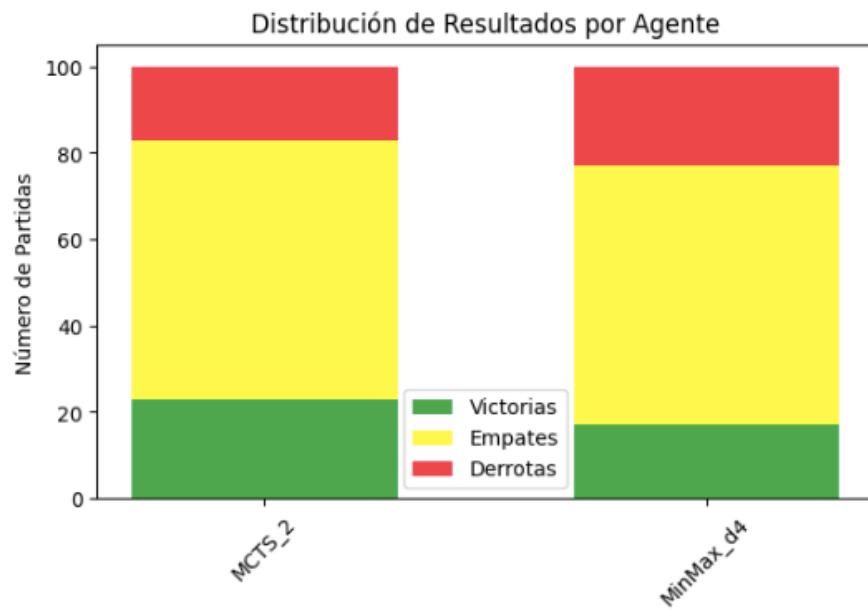
Evolución del Win Rate Acumulativo: Al inicio, MCTS\_2 obtiene mejores resultados, pero a medida que avanzan las partidas, ambos agentes muestran una tendencia a igualar sus desempeños, aunque MCTS mantiene una ligera ventaja.

### Tiempo promedio de decisión:



MCTS\_2 requiere un tiempo promedio de decisión ligeramente superior al de MinMax\_d4 (1.6 s frente a 1.3 s). Ambos tiempos son aceptables y no representan una diferencia significativa para el uso práctico.

### Distribución de Resultados por Agente



Ambos agentes muestran una gran proporción de empates (mayoría de las partidas). Sin embargo, MCTS\_2 logra una mayor cantidad de victorias y menos derrotas que MinMax\_d4, aunque la diferencia no es extrema.

### Conclusiones de los enfrentamientos:

El enfrentamiento directo entre los agentes MCTS\_2 y MinMax\_d4 en TicTacToe revela que MCTS tiene una ligera ventaja en términos de win rate y victorias directas, a costa de un pequeño incremento en el tiempo de cómputo por jugada. Sin embargo, la diferencia no es abrumadora y ambos agentes tienden a generar una gran cantidad de empates, lo que refleja la dificultad de superar consistentemente a un oponente bien calibrado en un juego determinístico y completamente observable como TicTacToe. En resumen, MCTS se posiciona como la estrategia ligeramente superior para este entorno, pero el margen de mejora respecto a MinMax es acotado.

#### *4.1.1.6 Resumen*

A lo largo de los experimentos realizados en TicTacToe se evaluaron y compararon distintas estrategias de decisión: MinMax, Monte Carlo Tree Search (MCTS) y agentes aleatorios, ajustando parámetros clave en cada caso.

- **MinMax:**

Se observó que el win rate del agente MinMax mejora al aumentar la profundidad del árbol de búsqueda, pero este incremento viene acompañado de un aumento significativo en el tiempo de decisión por jugada. La configuración óptima identificada fue una profundidad de 4, que ofrece un buen balance entre rendimiento y eficiencia computacional.

- **MCTS:**

El desempeño del agente MCTS depende críticamente del número de simulaciones y rollouts. El mejor trade-off se obtuvo con la configuración de 50 simulaciones y 20 rollouts (MCTS\_2), logrando un win rate alto y tiempos de decisión razonables. Aumentar excesivamente los parámetros sólo brinda mejoras marginales en rendimiento a cambio de un costo computacional mucho mayor.

- **Comparación contra Random:**

Tanto MinMax como MCTS superan ampliamente al agente aleatorio, con win rates superiores al 90% frente a menos del 5% para el Random. Esto confirma la superioridad de estrategias de búsqueda estructurada en juegos determinísticos y de información perfecta.

- **MCTS vs MinMax:**

El enfrentamiento directo muestra que MCTS\_2 tiene una ligera ventaja sobre MinMax\_d4 en términos de win rate y cantidad de victorias, aunque ambos agentes generan una alta proporción de empates. La diferencia de desempeño entre los dos enfoques es pequeña, destacando que ambos algoritmos, bien parametrizados, pueden lograr resultados muy competitivos.

En síntesis, en TicTacToe, tanto MinMax como MCTS se consolidan como estrategias robustas cuando sus parámetros se ajustan adecuadamente. El

balance entre profundidad de búsqueda o cantidad de simulaciones y el tiempo de decisión es clave para obtener un agente eficaz y eficiente. Finalmente, el margen de mejora entre los mejores agentes es limitado, y la mayoría de las partidas entre ellos tiende al empate, reflejando la naturaleza resuelta y simétrica del juego.

#### 4.1.2 Nocca Nocca

Se puede encontrar la notebook dentro de Juegos\_Alternados / Obligatorio / Nocca\_Nocca / Nocca\_Nocca.ipynb.

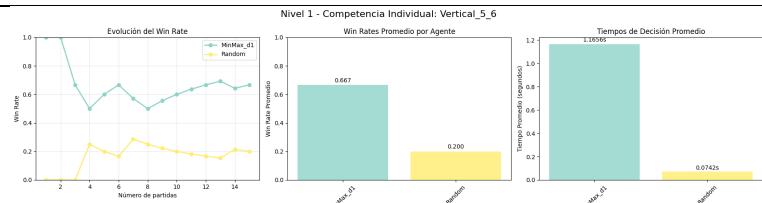
##### 4.1.2.1 MinMax depth=1 vs Random

###### Nivel 1:

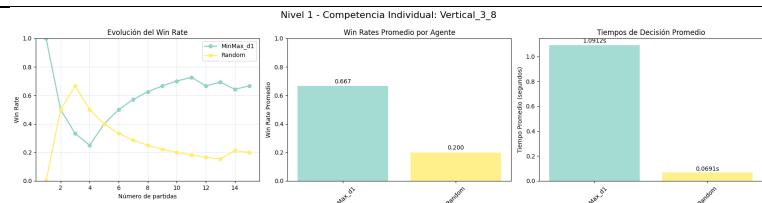
Los resultados para cada uno de los experimentos se ilustran en la tabla acontinuacion:

	$\alpha_D$	$\alpha_B$	Type	Grafico
<b>Vertical_10_0</b>	10	0	Vertical	<p><b>Nivel 1 - Competencia Individual: Vertical_10_0</b></p> <p>Evolución del Win Rate: Line chart showing Win Rate vs Number of games (2 to 14). MinMax_d1 (blue) starts at 1.0 and decreases to ~0.85; Random (orange) starts at 0.0 and increases to ~0.25.</p> <p>Win Rates Promedio por Agente: Bar chart showing Average Win Rate per Agent. MinMax_d1 is ~0.867; Random is ~0.133.</p> <p>Tiempos de Decisión Promedio: Bar chart showing Average Decision Time. MinMax_d1 is ~0.8901s; Random is ~0.0535s.</p>
<b>Vertical_8_3</b>	8	3	Vertical	<p><b>Nivel 1 - Competencia Individual: Vertical_8_3</b></p> <p>Evolución del Win Rate: Line chart showing Win Rate vs Number of games (2 to 14). MinMax_d1 (blue) starts at 1.0 and decreases to ~0.85; Random (orange) starts at 0.0 and increases to ~0.25.</p> <p>Win Rates Promedio por Agente: Bar chart showing Average Win Rate per Agent. MinMax_d1 is ~0.800; Random is ~0.200.</p> <p>Tiempos de Decisión Promedio: Bar chart showing Average Decision Time. MinMax_d1 is ~0.7123s; Random is ~0.0453s.</p>
<b>Vertical_6_5</b>	6	5	Vertical	<p><b>Nivel 1 - Competencia Individual: Vertical_6_5</b></p> <p>Evolución del Win Rate: Line chart showing Win Rate vs Number of games (2 to 14). MinMax_d1 (blue) starts at 1.0 and decreases to ~0.85; Random (orange) starts at 0.0 and increases to ~0.25.</p> <p>Win Rates Promedio por Agente: Bar chart showing Average Win Rate per Agent. MinMax_d1 is ~0.467; Random is ~0.533.</p> <p>Tiempos de Decisión Promedio: Bar chart showing Average Decision Time. MinMax_d1 is ~0.3689s; Random is ~0.0617s.</p>
<b>Vertical_5_5</b>	5	5	Vertical	<p><b>Nivel 1 - Competencia Individual: Vertical_5_5</b></p> <p>Evolución del Win Rate: Line chart showing Win Rate vs Number of games (2 to 14). MinMax_d1 (blue) starts at 0.0 and increases to ~0.55; Random (orange) starts at 1.0 and decreases to ~0.45.</p> <p>Win Rates Promedio por Agente: Bar chart showing Average Win Rate per Agent. MinMax_d1 is ~0.600; Random is ~0.400.</p> <p>Tiempos de Decisión Promedio: Bar chart showing Average Decision Time. MinMax_d1 is ~0.0124s; Random is ~0.0613s.</p>

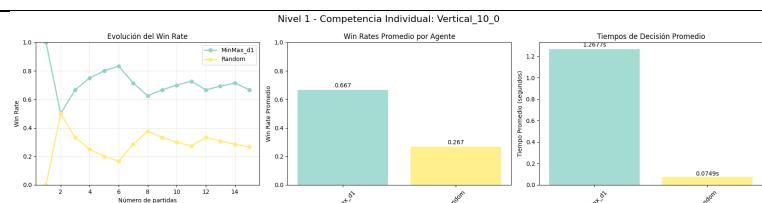
**Vertical\_5\_6**    5    6    Vertical



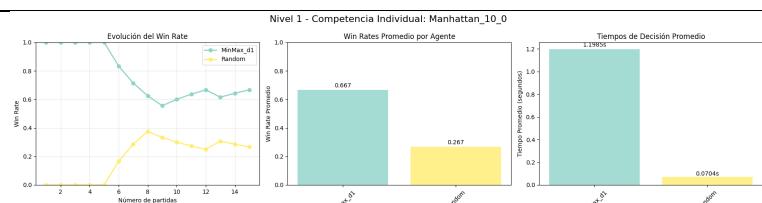
**Vertical\_3\_8**    3    8    Vertical



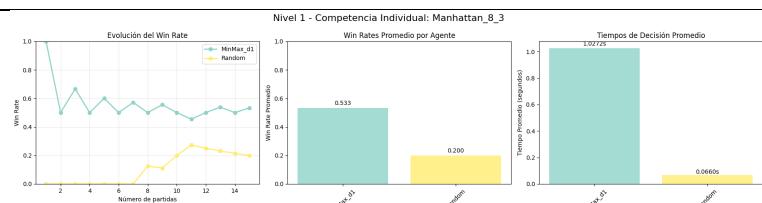
**Vertical\_0\_10**    0    10    Vertical



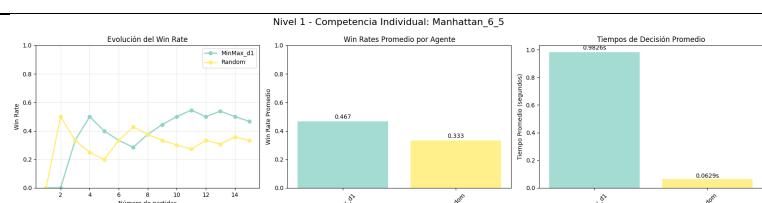
**Manhattan\_10\_0**    10    0    Manhattan



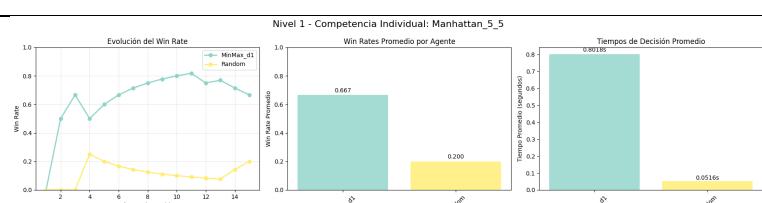
**Manhattan\_8\_3**    8    3    Manhattan



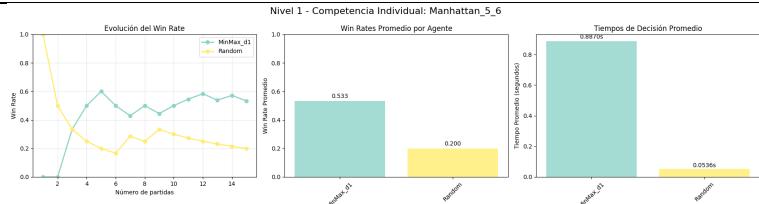
**Manhattan\_6\_5**    6    5    Manhattan



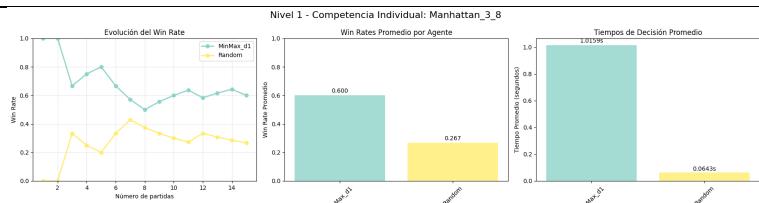
**Manhattan\_5\_5**    5    5    Manhattan



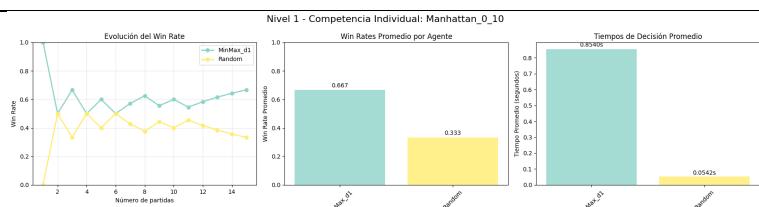
**Manhattan\_5\_6**    5    6    Manhattan



**Manhattan\_3\_8**    3    8    Manhattan

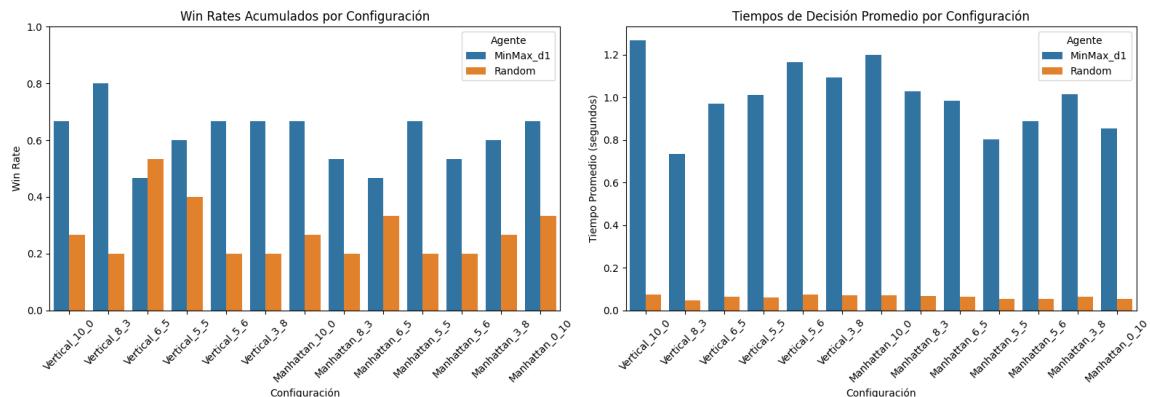


**Manhattan\_0\_10**    0    10    Manhattan



## Nivel 2:

Nivel 2 - Torneo General: Comparación entre Funciones de Evaluación



A partir de los gráficos presentados, se pueden extraer las siguientes conclusiones:

- Complejidad y tiempos de decisión: Un agente MinMax con profundidad 1 requiere aproximadamente 1 segundo para tomar una decisión. Esto pone en evidencia la mayor complejidad del entorno Nocca Nocca en comparación con

juegos más simples como TicTacToe, donde el tiempo de decisión para configuraciones similares es mucho menor.

2. Evaluación de la función de evaluación: Si bien los tiempos de decisión son razonables para todas las configuraciones probadas, aún es prematuro concluir si estos valores son óptimos o si la función de evaluación puede considerarse eficiente. Será necesario realizar experimentos adicionales y comparar con variantes de mayor profundidad o con otras funciones de evaluación para obtener un diagnóstico definitivo.
3. Robustez frente al azar: Los resultados muestran que, en general, un agente MinMax con profundidad 1 es capaz de superar a un agente Random en la mayoría de las configuraciones, independientemente de los pesos asignados a la función de evaluación. No obstante, existen excepciones —por ejemplo, en la configuración Vertical\_6\_5— donde el rendimiento disminuye y la brecha frente al azar se reduce.
4. Mejor trade-off rendimiento/tiempo: Analizando conjuntamente el win rate y el tiempo promedio de decisión, se concluye que la mejor relación entre desempeño y eficiencia computacional para el agente MinMax de profundidad 1 se obtiene con la función de evaluación basada en distancia vertical, específicamente con  $\alpha_D=8$  y  $\alpha_B=3$  (Vertical\_8\_3). Esta configuración maximiza el win rate sin aumentar significativamente el tiempo de cómputo.

#### 4.1.2.2 MinMax Depth=1 vs MinMax Depth=2

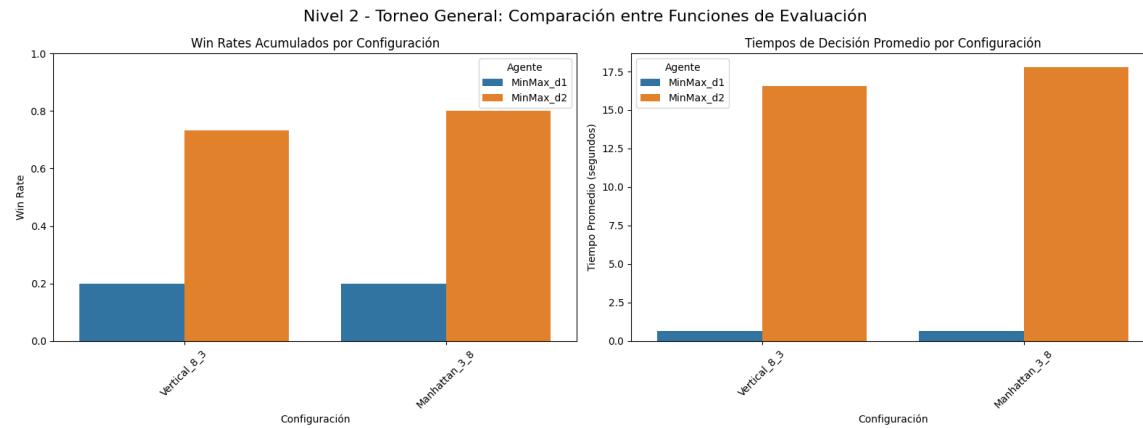
En este experimento se incorpora un agente MinMax de mayor profundidad, y se analiza su desempeño utilizando las dos mejores configuraciones para el tipo vertical y manhattan.

##### Nivel 1:

Los resultados para cada uno de los experimentos se ilustran en la tabla acontinuacion:

	$\alpha D$	$\alpha B$	Type	Grafico
<b>Vertical_8_3</b>	8	3	Vertical	<p>Nivel 1 - Competencia Individual: Vertical_8_3</p> <p>Evolución del Win Rate: Win Rate vs Number of matches (2 to 14). MinMax_d1 (blue) starts at ~0.55 and drops to ~0.15. MinMax_d2 (orange) starts at ~0.95 and rises to ~0.85.</p> <p>Win Rates Promedio por Agente: MinMax_d1 (0.200), MinMax_d2 (0.733).</p> <p>Tiempos de Decisión Promedio: MinMax_d1 (0.0513s), MinMax_d2 (17.314s).</p>
<b>Manhattan_3_8</b>	3	8	Manhattan	<p>Nivel 1 - Competencia Individual: Manhattan_3_8</p> <p>Evolución del Win Rate: Win Rate vs Number of matches (2 to 14). MinMax_d1 (blue) starts at ~0.55 and drops to ~0.15. MinMax_d2 (orange) starts at ~0.95 and rises to ~0.75.</p> <p>Win Rates Promedio por Agente: MinMax_d1 (0.290), MinMax_d2 (0.800).</p> <p>Tiempos de Decisión Promedio: MinMax_d1 (0.6350s), MinMax_d2 (17.771s).</p>

##### Nivel 2:



En esta configuración se observa claramente la ventaja de incrementar la profundidad del agente: con solo un nivel adicional, el desempeño (win rate)

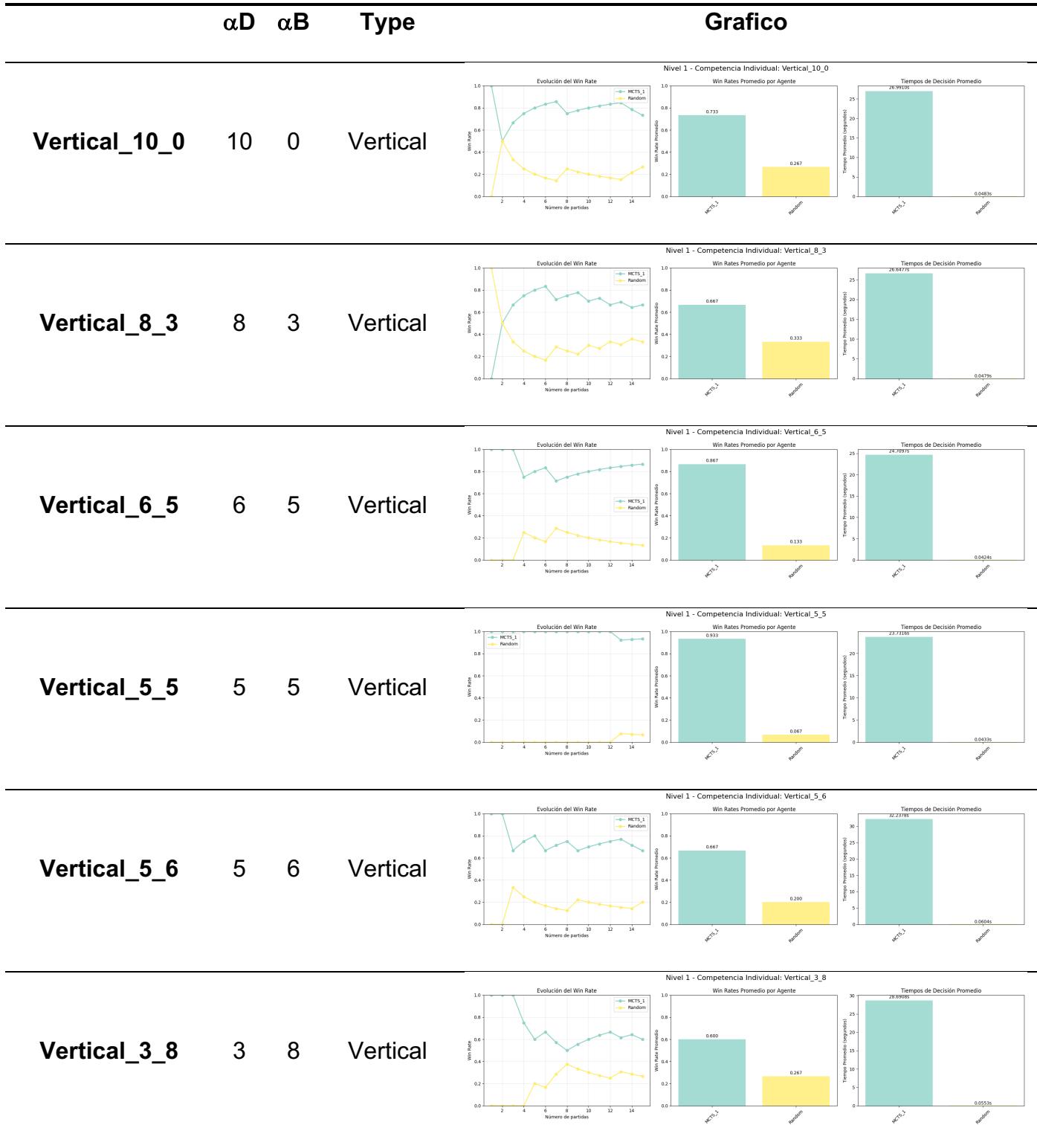
mejora significativamente respecto a la profundidad 1. Sin embargo, este avance tiene un costo computacional considerable, ya que el tiempo promedio de decisión se multiplica por más de 15 veces, superando los 16 segundos por jugada.

Por otro lado, al emplear un agente de mayor profundidad, la función de evaluación basada en distancia Manhattan —más compleja que la vertical— permite aprovechar mejor las posibilidades estratégicas del entorno, obteniendo un win rate ligeramente superior. No obstante, las diferencias entre Manhattan y vertical son marginales tanto en tiempo de ejecución como en desempeño cuando la profundidad es igual a 2.

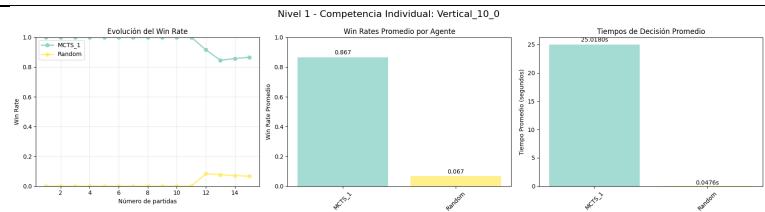
En síntesis, los resultados sugieren que aumentar la profundidad hace que el agente pueda beneficiarse de funciones de evaluación más sofisticadas, aunque el beneficio incremental se ve rápidamente limitado por el crecimiento del costo computacional.

#### 4.1.2.3 MCTS simulations=4 rollouts=2 vs Random

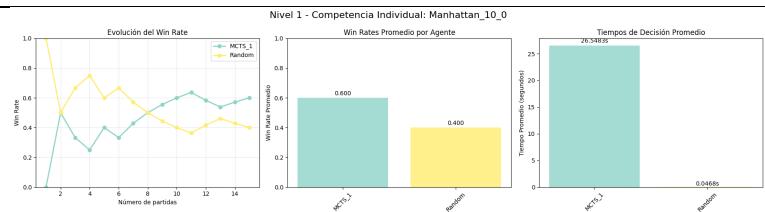
Nivel 1:



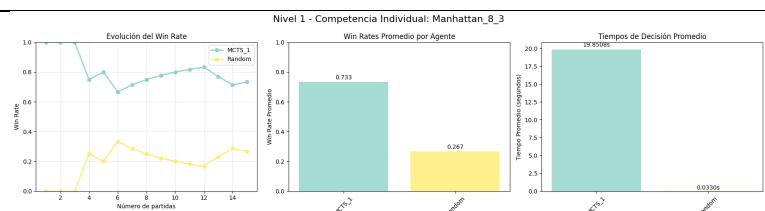
**Vertical\_0\_10** 0 10 Vertical



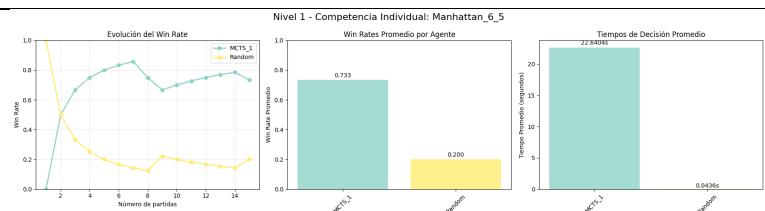
**Manhattan\_10\_0** 10 0 Manhattan



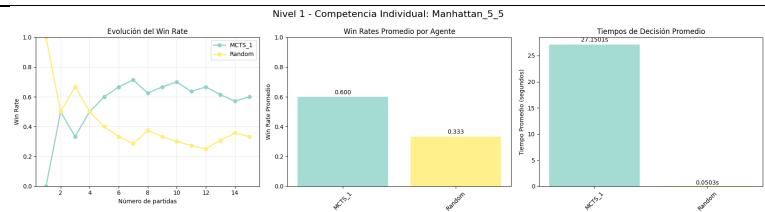
**Manhattan\_8\_3** 8 3 Manhattan



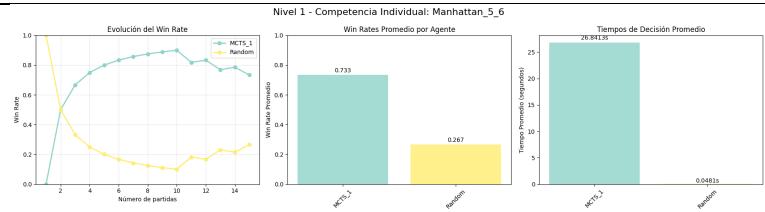
**Manhattan\_6\_5** 6 5 Manhattan



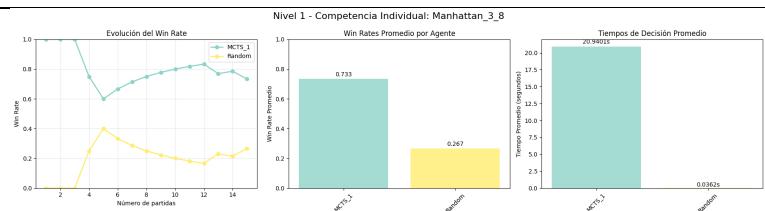
**Manhattan\_5\_5** 5 5 Manhattan



**Manhattan\_5\_6** 5 6 Manhattan

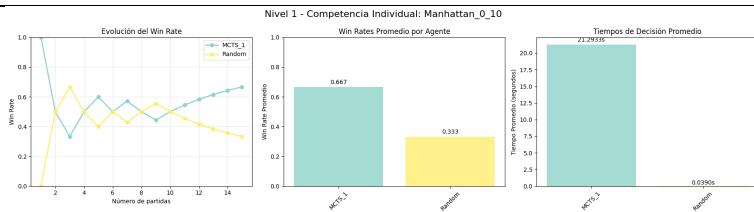


**Manhattan\_3\_8** 3 8 Manhattan



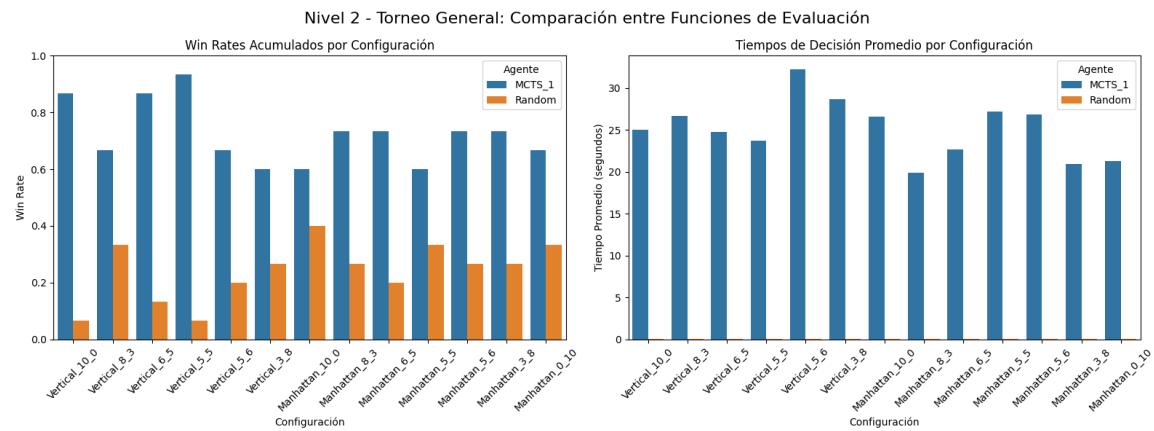
---

**Manhattan\_0\_10** 0 10 Manhattan



La tabla anterior resume los resultados de cada experimento realizado con las diferentes configuraciones de la función de evaluación (variando  $\alpha_D$ ,  $\alpha_B$  y el tipo de distancia: vertical o Manhattan). En todas las combinaciones, el agente MCTS se enfrentó a un agente Random.

### Nivel 2:



A partir de los gráficos presentados, se pueden extraer las siguientes conclusiones:

1. Complejidad y tiempos de decisión: Un agente MCTS configurado con solo 4 simulaciones y 2 rollouts por jugada —es decir, una configuración considerablemente más sencilla que las utilizadas en TicTacToe— logra superar con claridad al agente Random en todas las variantes de la función de evaluación. Sin embargo, este desempeño viene acompañado de un costo computacional muy elevado en comparación con MinMax, reflejando tanto la mayor complejidad estratégica del entorno Nocca Nocca como el mayor poder de exploración y búsqueda de MCTS..
2. Evaluación de la función de evaluación: Aunque podría pensarse que funciones de evaluación más costosas, como la distancia Manhattan,

incrementarían significativamente el tiempo de decisión, los resultados muestran que el factor determinante es la dinámica interna del algoritmo y la complejidad del entorno, más que el tipo de función de evaluación utilizada. Incluso en los casos extremos (por ejemplo,  $\alpha_D=0$  y  $\alpha_B=10$ ), el tiempo de decisión y los resultados no varían significativamente respecto a otras configuraciones.

No obstante, se observan ciertas diferencias durante el desarrollo de las partidas:

- **Vertical\_0\_10:** Win Rate de 0.867 y tiempo promedio de ~25 s.
- **Manhattan\_0\_10:** Win Rate de 0.667 y tiempo promedio de ~21 s.

Dado que el componente de distancia está anulado, estas diferencias se atribuyen a la implementación: aunque la función de distancia no afecta el valor final de la evaluación, el algoritmo igualmente ejecuta el recorrido correspondiente (ya sea Manhattan o vertical), lo que puede influir en el orden de exploración de nodos y en el costo computacional. Además, hay que considerar el ruido estocástico propio de MCTS y el hecho de que la cantidad de partidas jugadas en cada experimento no es muy alta.

3. Robustez frente al azar: Los resultados reafirman que el agente MCTS, incluso en configuraciones modestas, es consistentemente superior al agente Random, manteniendo un win rate alto y estable en todas las variantes de la función de evaluación. Esto pone en evidencia la robustez de MCTS frente a decisiones aleatorias, independientemente de los pesos utilizados.
4. Mejor trade-off rendimiento/tiempo: Al analizar el balance entre win rate y tiempo de decisión, la mejor relación entre rendimiento y eficiencia computacional para el agente MCTS se observa en la configuración Vertical\_5\_5 ( $\alpha_D=5$ ,  $\alpha_B=5$ , distancia vertical). Esta opción permite mantener un win rate elevado sin incrementar significativamente el tiempo requerido para decidir cada jugada.

#### 4.1.2.4 MCTS vs MCTS

En este apartado se compara el desempeño de un agente MCTS con la misma configuración utilizada previamente (simulations=4, rollouts=2) frente a un agente MCTS ligeramente más complejo, configurado con simulations=8 y rollouts=4. El objetivo es analizar cómo impacta este aumento moderado en la cantidad de simulaciones y rollouts tanto en el rendimiento (win rate) como en el tiempo de decisión por jugada.

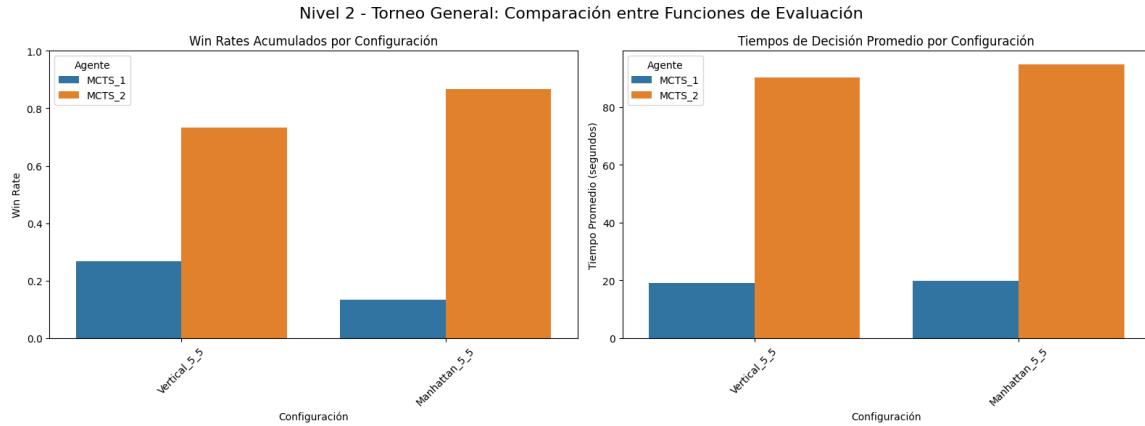
##### Nivel 1:

Los resultados para cada experimento se resumen en la siguiente tabla, considerando dos variantes de la función de evaluación:

- Vertical\_5\_5: ( $\alpha_D=5$ ,  $\alpha_B=5$ , distancia vertical)
- Manhattan\_5\_5: ( $\alpha_D=5$ ,  $\alpha_B=5$ , distancia Manhattan)

	$\alpha D$	$\alpha B$	Type	Grafico																																				
<b>Vertical_5_5</b>	5	5	Vertical	<p>Nivel 1 - Competencia Individual: Vertical_5_5</p> <p>Evolución del Win Rate</p> <table border="1"> <thead> <tr> <th>Partida</th> <th>MCTS_1</th> <th>MCTS_2</th> </tr> </thead> <tbody> <tr><td>2</td><td>0.0</td><td>1.0</td></tr> <tr><td>4</td><td>0.2</td><td>0.8</td></tr> <tr><td>6</td><td>0.15</td><td>0.75</td></tr> <tr><td>8</td><td>0.15</td><td>0.8</td></tr> <tr><td>10</td><td>0.15</td><td>0.85</td></tr> <tr><td>12</td><td>0.15</td><td>0.85</td></tr> <tr><td>14</td><td>0.25</td><td>0.75</td></tr> </tbody> </table> <p>Win Rates Promedio por Agente</p> <table border="1"> <thead> <tr> <th>Agente</th> <th>Win Rate Promedio</th> </tr> </thead> <tbody> <tr><td>MCTS_1</td><td>0.267</td></tr> <tr><td>MCTS_2</td><td>0.733</td></tr> </tbody> </table> <p>Tiempos de Decisión Promedio</p> <table border="1"> <thead> <tr> <th>Agente</th> <th>Tiempo de Decisión Promedio (ms)</th> </tr> </thead> <tbody> <tr><td>MCTS_1</td><td>18.9529s</td></tr> <tr><td>MCTS_2</td><td>90.3708s</td></tr> </tbody> </table>	Partida	MCTS_1	MCTS_2	2	0.0	1.0	4	0.2	0.8	6	0.15	0.75	8	0.15	0.8	10	0.15	0.85	12	0.15	0.85	14	0.25	0.75	Agente	Win Rate Promedio	MCTS_1	0.267	MCTS_2	0.733	Agente	Tiempo de Decisión Promedio (ms)	MCTS_1	18.9529s	MCTS_2	90.3708s
Partida	MCTS_1	MCTS_2																																						
2	0.0	1.0																																						
4	0.2	0.8																																						
6	0.15	0.75																																						
8	0.15	0.8																																						
10	0.15	0.85																																						
12	0.15	0.85																																						
14	0.25	0.75																																						
Agente	Win Rate Promedio																																							
MCTS_1	0.267																																							
MCTS_2	0.733																																							
Agente	Tiempo de Decisión Promedio (ms)																																							
MCTS_1	18.9529s																																							
MCTS_2	90.3708s																																							
<b>Manhattan_5_5</b>	5	5	Manhattan	<p>Nivel 1 - Competencia Individual: Manhattan_5_5</p> <p>Evolución del Win Rate</p> <table border="1"> <thead> <tr> <th>Partida</th> <th>MCTS_1</th> <th>MCTS_2</th> </tr> </thead> <tbody> <tr><td>2</td><td>0.0</td><td>1.0</td></tr> <tr><td>4</td><td>0.0</td><td>0.8</td></tr> <tr><td>6</td><td>0.0</td><td>0.8</td></tr> <tr><td>8</td><td>0.0</td><td>0.8</td></tr> <tr><td>10</td><td>0.0</td><td>0.8</td></tr> <tr><td>12</td><td>0.1</td><td>0.8</td></tr> <tr><td>14</td><td>0.15</td><td>0.8</td></tr> </tbody> </table> <p>Win Rates Promedio por Agente</p> <table border="1"> <thead> <tr> <th>Agente</th> <th>Win Rate Promedio</th> </tr> </thead> <tbody> <tr><td>MCTS_1</td><td>0.133</td></tr> <tr><td>MCTS_2</td><td>0.867</td></tr> </tbody> </table> <p>Tiempos de Decisión Promedio</p> <table border="1"> <thead> <tr> <th>Agente</th> <th>Tiempo de Decisión Promedio (ms)</th> </tr> </thead> <tbody> <tr><td>MCTS_1</td><td>19.7082s</td></tr> <tr><td>MCTS_2</td><td>94.7955s</td></tr> </tbody> </table>	Partida	MCTS_1	MCTS_2	2	0.0	1.0	4	0.0	0.8	6	0.0	0.8	8	0.0	0.8	10	0.0	0.8	12	0.1	0.8	14	0.15	0.8	Agente	Win Rate Promedio	MCTS_1	0.133	MCTS_2	0.867	Agente	Tiempo de Decisión Promedio (ms)	MCTS_1	19.7082s	MCTS_2	94.7955s
Partida	MCTS_1	MCTS_2																																						
2	0.0	1.0																																						
4	0.0	0.8																																						
6	0.0	0.8																																						
8	0.0	0.8																																						
10	0.0	0.8																																						
12	0.1	0.8																																						
14	0.15	0.8																																						
Agente	Win Rate Promedio																																							
MCTS_1	0.133																																							
MCTS_2	0.867																																							
Agente	Tiempo de Decisión Promedio (ms)																																							
MCTS_1	19.7082s																																							
MCTS_2	94.7955s																																							

## Nivel 2:



En esta comparación, al igual que en los experimentos previos con MinMax, se observa una clara mejora en el rendimiento al incrementar la “profundidad” del agente, es decir, la cantidad de simulaciones y rollouts. Sin embargo, esta ganancia viene acompañada de un costo computacional muy alto: el tiempo promedio de decisión puede superar los 80 segundos por jugada.

Por otro lado, emplear funciones de evaluación más sofisticadas, como Manhattan, tiende a dar mejores resultados que la función vertical.

En síntesis, los resultados muestran que incrementar la capacidad de exploración del agente MCTS puede mejorar su rendimiento y explotar mejor funciones de evaluación complejas. No obstante, el beneficio incremental se ve rápidamente opacado por el crecimiento exponencial del tiempo de cómputo, por lo que encontrar un buen equilibrio entre profundidad y eficiencia sigue siendo fundamental en este tipo de entornos.

#### 4.1.2.5 MCTS vs MinMax

En esta sección se compara el desempeño de un agente MCTS frente a un agente MinMax, empleando versiones simplificadas de ambos algoritmos y limitando la cantidad de partidas para evitar experimentos excesivamente largos.

Configuraciones utilizadas:

- MCTS\_1: 50 simulaciones, 10 rollouts
- MinMax\_1: profundidad = 1

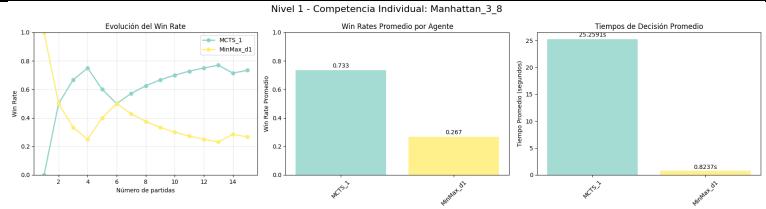
##### Nivel 1

Los resultados para cada experimento se resumen en la siguiente tabla:

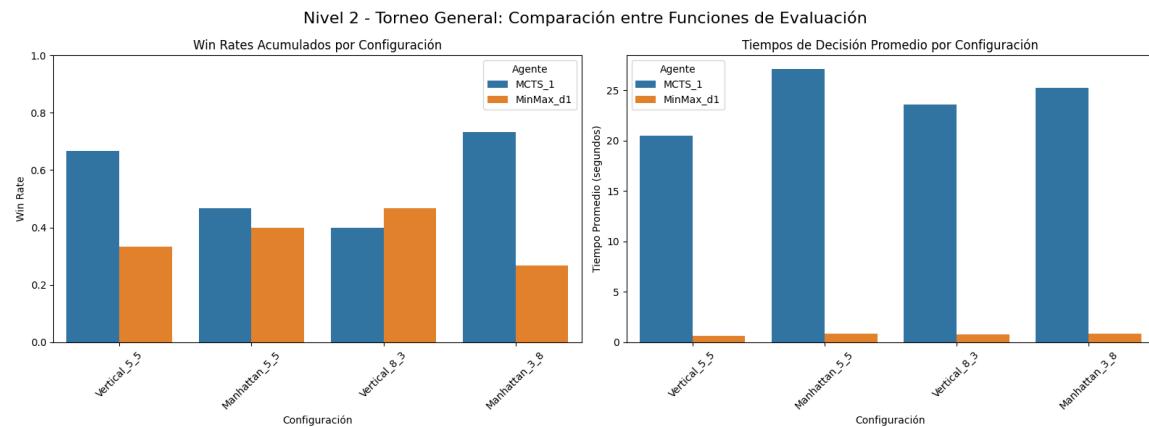
- Vertical\_5\_5: ( $\alpha_D=5$ ,  $\alpha_B=5$ , distancia vertical)
- Manhattan\_5\_5: ( $\alpha_D=5$ ,  $\alpha_B=5$ , distancia Manhattan)
- Vertical\_8\_3: ( $\alpha_D=8$ ,  $\alpha_B=3$ , distancia vertical)
- Manhattan\_3\_8: ( $\alpha_D=3$ ,  $\alpha_B=8$ , distancia Manhattan)

	$\alpha D$	$\alpha B$	Type	Grafico												
<b>Vertical_5_5</b>	5	5	Vertical	<p>Nivel 1 - Competencia Individual: Vertical_5_5</p> <table border="1"> <thead> <tr> <th>Agente</th> <th>Win Rate Promedio</th> </tr> </thead> <tbody> <tr> <td>MCTS_1</td> <td>0.667</td> </tr> <tr> <td>MinMax_1</td> <td>0.333</td> </tr> </tbody> </table> <table border="1"> <thead> <tr> <th>Agente</th> <th>Tiempo de Decisión Promedio (ms)</th> </tr> </thead> <tbody> <tr> <td>MCTS_1</td> <td>20.30955</td> </tr> <tr> <td>MinMax_1</td> <td>0.64735</td> </tr> </tbody> </table>	Agente	Win Rate Promedio	MCTS_1	0.667	MinMax_1	0.333	Agente	Tiempo de Decisión Promedio (ms)	MCTS_1	20.30955	MinMax_1	0.64735
Agente	Win Rate Promedio															
MCTS_1	0.667															
MinMax_1	0.333															
Agente	Tiempo de Decisión Promedio (ms)															
MCTS_1	20.30955															
MinMax_1	0.64735															
<b>Manhattan_5_5</b>	5	5	Manhattan	<p>Nivel 1 - Competencia Individual: Manhattan_5_5</p> <table border="1"> <thead> <tr> <th>Agente</th> <th>Win Rate Promedio</th> </tr> </thead> <tbody> <tr> <td>MCTS_1</td> <td>0.467</td> </tr> <tr> <td>MinMax_1</td> <td>0.400</td> </tr> </tbody> </table> <table border="1"> <thead> <tr> <th>Agente</th> <th>Tiempo de Decisión Promedio (ms)</th> </tr> </thead> <tbody> <tr> <td>MCTS_1</td> <td>27.11145</td> </tr> <tr> <td>MinMax_1</td> <td>0.85095</td> </tr> </tbody> </table>	Agente	Win Rate Promedio	MCTS_1	0.467	MinMax_1	0.400	Agente	Tiempo de Decisión Promedio (ms)	MCTS_1	27.11145	MinMax_1	0.85095
Agente	Win Rate Promedio															
MCTS_1	0.467															
MinMax_1	0.400															
Agente	Tiempo de Decisión Promedio (ms)															
MCTS_1	27.11145															
MinMax_1	0.85095															
<b>Vertical_8_3</b>	8	3	Vertical	<p>Nivel 1 - Competencia Individual: Vertical_8_3</p> <table border="1"> <thead> <tr> <th>Agente</th> <th>Win Rate Promedio</th> </tr> </thead> <tbody> <tr> <td>MCTS_1</td> <td>0.400</td> </tr> <tr> <td>MinMax_1</td> <td>0.467</td> </tr> </tbody> </table> <table border="1"> <thead> <tr> <th>Agente</th> <th>Tiempo de Decisión Promedio (ms)</th> </tr> </thead> <tbody> <tr> <td>MCTS_1</td> <td>23.57425</td> </tr> <tr> <td>MinMax_1</td> <td>0.75165</td> </tr> </tbody> </table>	Agente	Win Rate Promedio	MCTS_1	0.400	MinMax_1	0.467	Agente	Tiempo de Decisión Promedio (ms)	MCTS_1	23.57425	MinMax_1	0.75165
Agente	Win Rate Promedio															
MCTS_1	0.400															
MinMax_1	0.467															
Agente	Tiempo de Decisión Promedio (ms)															
MCTS_1	23.57425															
MinMax_1	0.75165															

**Manhattan\_3\_8**    3    8    Manhattan



## Nivel 2:



Los resultados muestran que MCTS es, en general, una estrategia más robusta y adaptable, logrando superar a MinMax en la mayoría de las variantes de la función de evaluación, incluso con una configuración de baja complejidad. La excepción a esta tendencia se da en la configuración Vertical\_5\_5, donde MinMax depth=1 obtiene su mejor desempeño y logra imponerse frente a MCTS.

No obstante, en lo que respecta al tiempo de decisión, MCTS sigue presentando costos computacionales significativamente más altos en comparación con MinMax, lo que refuerza que el mejor trade-off entre rendimiento y eficiencia lo alcanza el agente MinMax en la configuración Vertical\_5\_5.

En resumen, aunque MCTS demuestra ser más flexible y efectivo en la mayoría de los escenarios, MinMax puede ser preferible en entornos donde los recursos computacionales y el tiempo de respuesta sean críticos, especialmente utilizando una función de evaluación bien ajustada.

#### *4.1.2.6 Resumen*

A lo largo de los experimentos realizados en Nocca Nocca, se observaron las siguientes conclusiones principales:

- **MCTS vs MinMax:**

MCTS es un algoritmo considerablemente más complejo, capaz de encontrar estrategias superiores y adaptarse mejor a diferentes configuraciones de la función de evaluación. Sin embargo, su costo computacional es alto y no siempre justifica su uso en entornos donde los recursos o el tiempo de respuesta sean críticos. En cambio, MinMax, siendo un algoritmo eficiente y confiable para profundidades bajas, aunque su alcance está determinado por el límite práctico de la profundidad de búsqueda, a diferencia de MCTS, que explora de forma más amplia gracias a su naturaleza estocástica y flexible, puede lograr un excelente desempeño con funciones de evaluación bien ajustadas y profundidades bajas, alcanzando el mejor balance entre win rate y eficiencia en configuraciones como Vertical\_5\_5.

- **Importancia de la función de evaluación:**

La función de evaluación desarrollada cumple con los requisitos básicos de distinguir entre estados, facilitar el desarrollo de estrategias efectivas y ser computacionalmente manejable para búsquedas de baja profundidad. Sin embargo, a medida que aumenta la complejidad del entorno y de los agentes, su aporte marginal a la mejora del desempeño se reduce frente a los límites de hardware o tiempo.

- **Rendimiento, Robustez y Trade-off:**

MCTS muestra robustez frente al azar y es consistente en superar agentes Random incluso con configuraciones simples, destacando su capacidad de exploración estratégica.

MinMax es competitivo y eficiente, especialmente cuando se prioriza la velocidad de decisión y el entorno no permite búsquedas profundas.

El mejor trade-off entre rendimiento y tiempo se alcanza usando agentes MinMax de baja profundidad junto a funciones de evaluación vertical bien calibradas.

- Complejidad del entorno Nocca Nocca:

Se evidencia que Nocca Nocca, por sus reglas y posibilidades estratégicas, es un entorno mucho más exigente computacionalmente que juegos clásicos como TicTacToe. Esto obliga a ser cuidadoso al seleccionar algoritmos y configuraciones, ya que pequeñas variaciones pueden multiplicar el tiempo de cómputo.

- Conclusiones para elegir agente:

- Si el objetivo es maximizar la performance y se dispone de tiempo/recursos, MCTS con suficientes simulaciones y una función de evaluación balanceada con distancia manhattan es preferible.
- Si se necesita responder en tiempo real o el entorno restringe el poder de cómputo, MinMax con una función de evaluación vertical bien ajustada representa la opción óptima.

En ambos casos, experimentar con los parámetros de la función de evaluación (pesos  $\alpha_D$  y  $\alpha_B$ , y tipo de distancia) puede marcar la diferencia entre un agente competitivo y uno mediocre.

- Notas finales:

La sintonía entre la función de evaluación, la profundidad de búsqueda y el tipo de algoritmo es determinante. No existe una única solución óptima; la selección ideal depende tanto del objetivo (rendimiento, robustez, eficiencia) como de las restricciones prácticas del entorno donde se implementa el agente.

## 4.2 Juegos alternados de información imperfecta

Se puede encontrar las notebooks dentro de:

Juegos alternados/Obligatorio/Khun\_Poker/Khun\_poker\_2ipynb

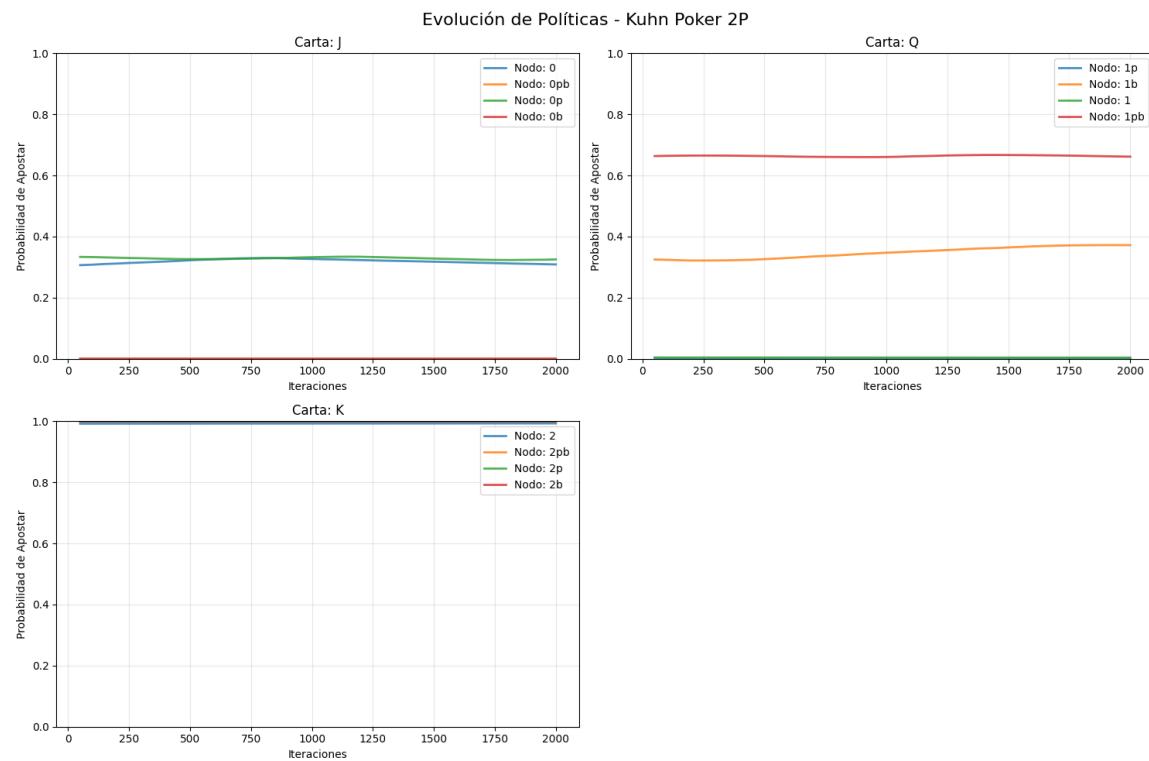
Juegos alternados/Obligatorio/Khun\_Poker/Khun\_poker\_3ipynb

### 4.2.1 Khun poker 2 jugadores

En este entorno se analiza el desempeño de un agente basado en *Counterfactual Regret Minimization* (CFR), evaluando tanto el efecto del número de iteraciones de entrenamiento como la influencia de la duración de las partidas en su comportamiento estratégico.

#### 4.2.1.1 Entrenamiento corto (10 iteraciones) y Corrida de estudio de 2000 partidas:

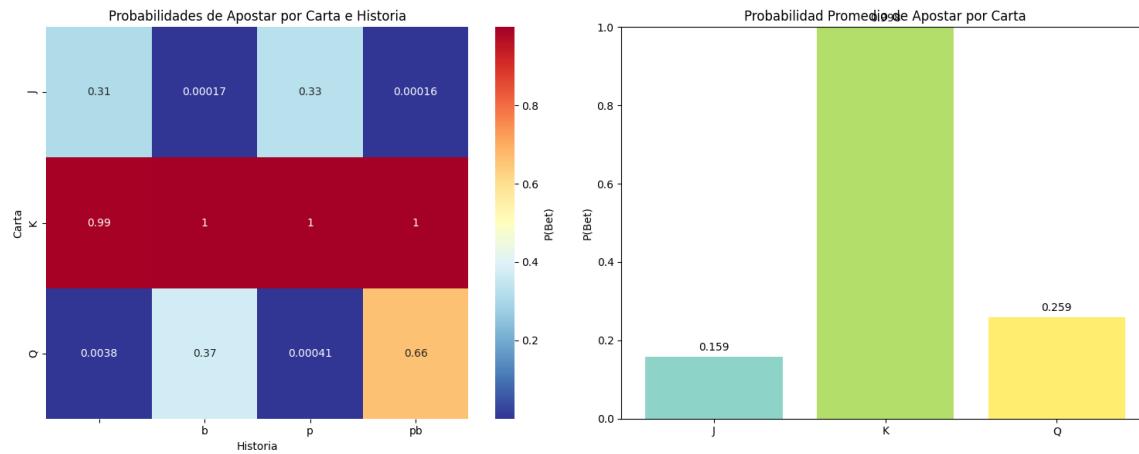
Evolucion de la politica:



Las curvas que representan la evolución de la probabilidad de apostar en cada nodo son en su mayoría planas o muestran oscilaciones muy leves. Este comportamiento es coherente con el bajo número de iteraciones de

entrenamiento. Al no haber pasado por suficientes rondas de actualización, el agente apenas ha comenzado a ajustar sus decisiones. A pesar de ello, se observa que en los nodos donde el agente posee la carta K, la probabilidad de apostar se estabiliza rápidamente en valores cercanos a 1. Esto indica que, incluso con muy poco entrenamiento, el agente reconoce la fortaleza de esa carta y adopta un comportamiento agresivo al respecto. En cambio, con cartas como J o Q, la política se muestra menos estructurada y más errática, sin patrones claros aún definidos.

### Heatmap de decisiones y Promedio de probabilidad de apostar por carta:



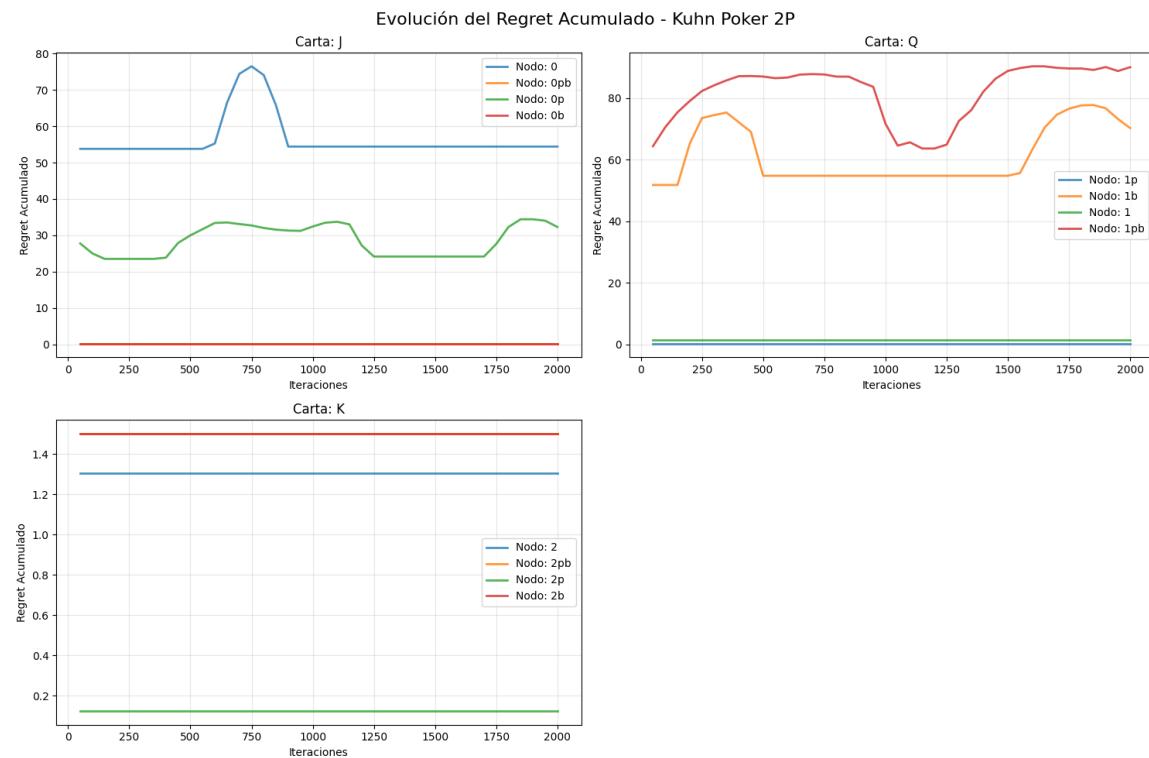
El mapa de calor refleja de manera compacta lo observado en la evolución de políticas. La carta K se asocia de forma sistemática con decisiones de apostar en prácticamente todos los históricos posibles, lo cual está alineado con el comportamiento óptimo en Kuhn Poker. Por otro lado, las decisiones tomadas con las cartas J y Q muestran una mayor dispersión: algunas combinaciones de carta e histórico presentan una alta probabilidad de apostar, mientras que otras apenas superan valores cercanos a cero. Esta inestabilidad es indicativa de que el agente aún no ha aprendido una estrategia consistente para las situaciones intermedias o de debilidad relativa.

El análisis de la política promedio en función de la carta obtenida permite resumir el estado general del comportamiento aprendido:

- Para la carta K, la probabilidad promedio de apostar es de 0.99, lo cual está perfectamente alineado con el equilibrio de Nash, donde se espera que el jugador actúe agresivamente con su mejor carta.
- Para la carta Q, se obtiene un valor de 0.25. Este valor sugiere que el agente comienza a implementar una estrategia mixta, que es adecuada para cartas de valor medio, aunque todavía no de manera sistemática.
- En el caso de la carta J, la probabilidad es de 0.15, lo cual resulta ligeramente elevado para lo que sería un bluff ocasional, pero sigue siendo aceptable teniendo en cuenta el escaso número de iteraciones.

Estos promedios, si bien no alcanzan el equilibrio, reflejan un ordenamiento razonable de las cartas en términos de agresividad, incluso en etapas tempranas del entrenamiento.

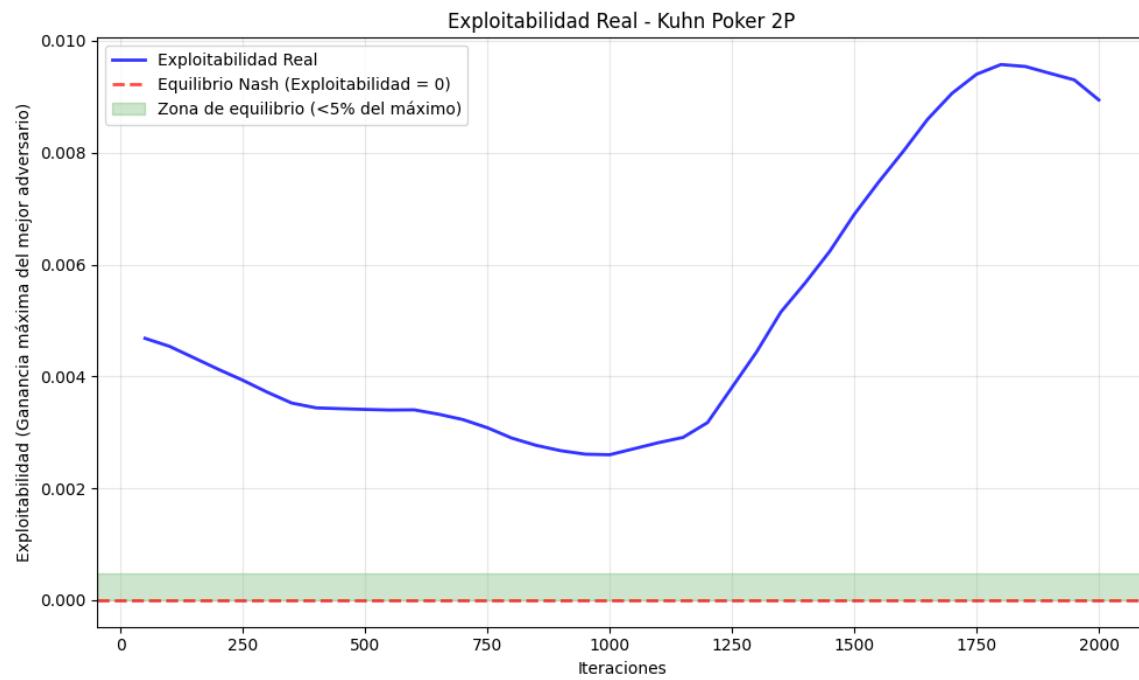
#### Evolución del regret acumulado:



Los registros de regret muestran que algunos nodos presentan picos de valor en las primeras mil partidas, especialmente aquellos asociados a las cartas J y Q en

situaciones de decisión intermedia. Posteriormente, dichos valores tienden a estabilizarse, aunque sin alcanzar una disminución significativa. En contraste, varios nodos muestran regret cercano a cero durante toda la simulación, particularmente aquellos asociados a secuencias menos frecuentes o terminales (por ejemplo, 2pb o 0pb). Esta distribución del regret indica que el agente comienza a identificar qué acciones resultan subóptimas en determinados contextos, pero no ha recorrido aún el espacio de juego con la suficiente profundidad como para ajustar con precisión su estrategia en todos los nodos relevantes.

### Explotabilidad real

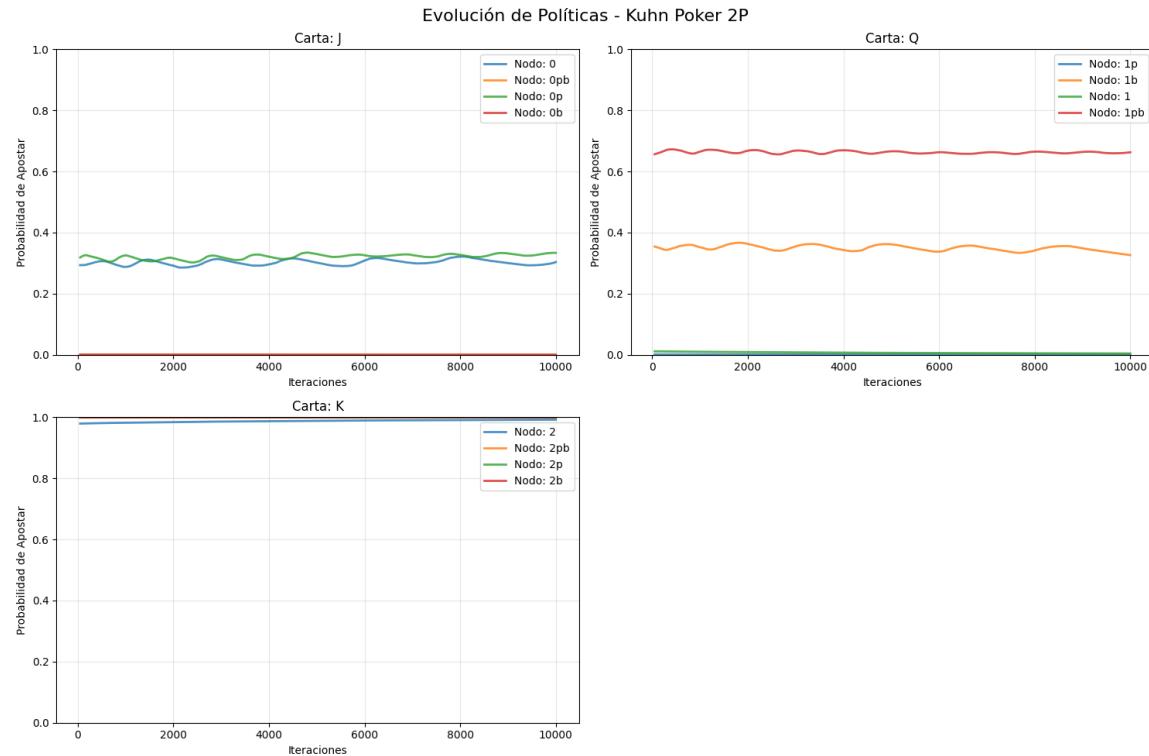


La evaluación de la explotabilidad real, entendida como la ganancia que podría obtener un oponente óptimo contra la política promedio aprendida, muestra un comportamiento interesante. La explotabilidad comienza en valores bajos (cerca de 0.0045) y alcanza un mínimo alrededor de las 1000 partidas simuladas. A partir de ese punto, comienza a incrementarse progresivamente, acercándose a valores del orden de 0.0095 hacia el final de las 2000 partidas. Esto sugiere que, si bien la política inicial contiene elementos de mezcla que la

hacen relativamente poco predecible, la falta de entrenamiento adicional impide su consolidación. A medida que el agente repite patrones no refinados, se vuelve más vulnerable frente a oponentes que pueden detectar y explotar sus debilidades.

#### 4.2.1.2 Entrenamiento corto (10 iteraciones) y Corrida de estudio de 10.000 partidas

##### Evolucion de la politica:

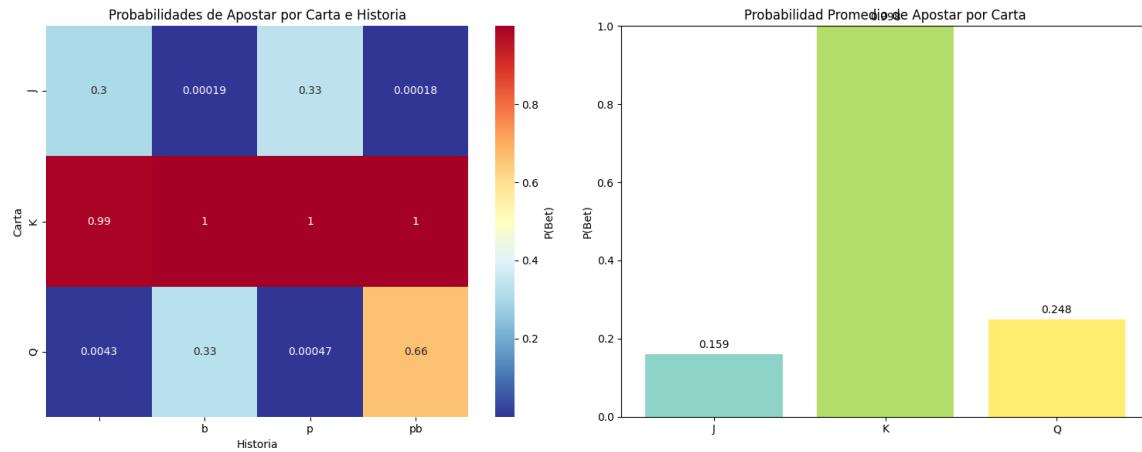


Las curvas de probabilidad por nodo se mantienen mayormente estables a lo largo de las 10.000 partidas, con oscilaciones leves. Esto es coherente con el hecho de que el entrenamiento fue extremadamente breve (solo 10 iteraciones), lo que limita la capacidad del agente de explorar y ajustar sus estrategias.

- Con carta K, se observa una apuesta prácticamente constante (cercana a 1), reflejando una elección agresiva que puede surgir incluso sin aprendizaje, dada la superioridad evidente de la K.
- Con carta Q, las probabilidades son más moderadas y oscilan entre nodos, reflejando un comportamiento mixto sin una convergencia clara.
- Con carta J, el agente mantiene políticas bajas de apuesta, lo cual es coherente con la debilidad de esa mano, aunque podría deberse más a decisiones iniciales aleatorias que a una política aprendida.

En resumen, las políticas reflejan patrones iniciales razonables más que un aprendizaje real, y su estabilidad se explica por la falta de entrenamiento posterior que permite su evolución.

#### Heatmap de decisiones y Promedio de probabilidad de apostar por carta:



El mapa de calor confirma una fuerte preferencia por apostar con K, con valores de 0.99 a 1 en todos los contextos posibles. Esto es esperable incluso sin entrenamiento, ya que se trata de la mejor carta del juego.

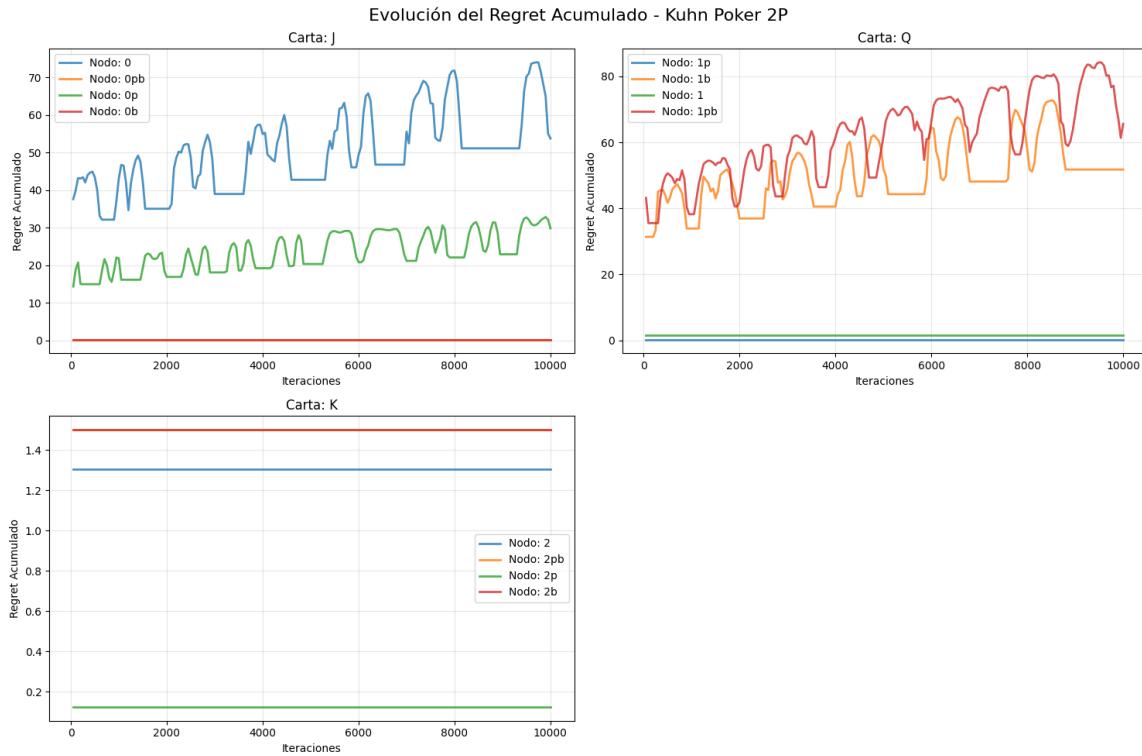
En contraste:

- Con Q, las decisiones de apuesta varían según el nodo, con valores altos en “pb” (0.66) y medios en “b” (0.33), lo cual indica cierta inclinación a presionar en contextos donde el rival podría tener J.
- Con J, las políticas de apuesta son bajas (0.30–0.33 en nodos de apertura), y cercanas a cero en otros, lo que sugiere una actitud conservadora.

Promedios generales de apuesta por carta:

- K: 0.99; comportamiento óptimo, incluso sin aprendizaje.
- Q: 0.25; valor intermedio que puede responder a decisiones aleatorias iniciales.
- J: 0.15; baja probabilidad, razonable por la debilidad de la mano, pero sin justificación táctica sólida a esta altura.

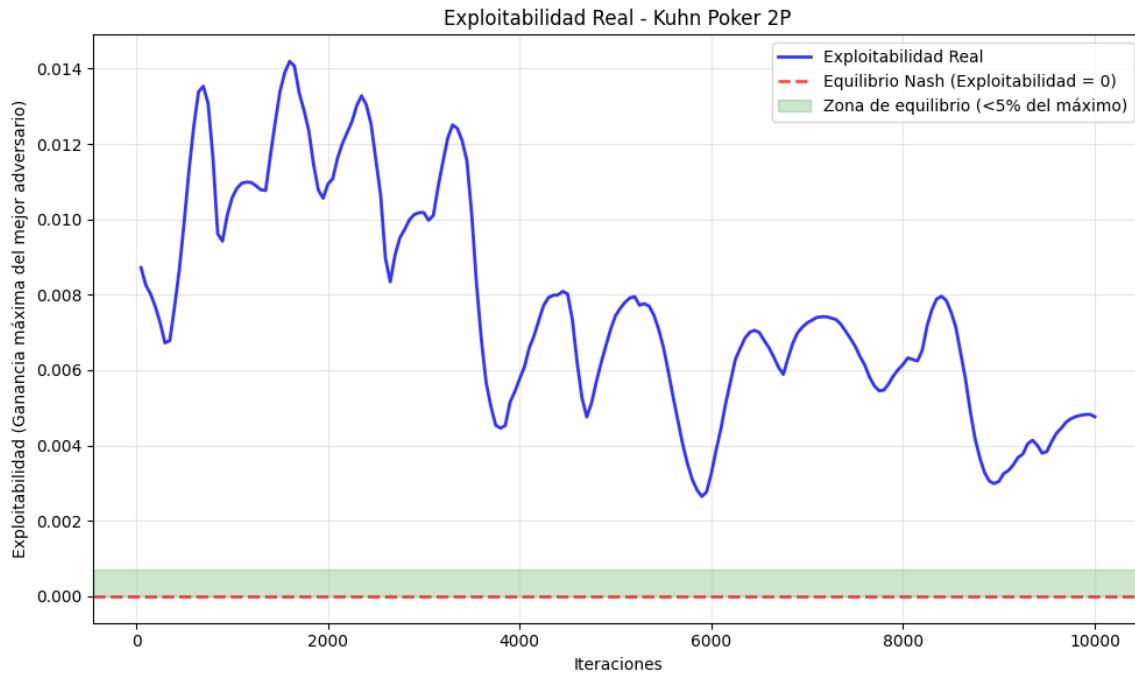
- Evolución del regret acumulado:



Se observa un comportamiento oscilatorio persistente en los nodos asociados a las cartas **J** y **Q**, lo cual indica que el agente aún no ha logrado estabilizar sus políticas en estas decisiones. Estas oscilaciones reflejan una fase de exploración activa, sin una convergencia clara hacia una estrategia dominante. En cambio, los nodos correspondientes a la carta **K** muestran un regret acumulado prácticamente constante y cercano a cero, lo que evidencia una elección sistemática y alineada con la estrategia óptima.

Al comparar estos resultados con los obtenidos tras 2000 partidas, se concluye que un mayor número de partidas de evaluación no compensa la falta de entrenamiento previo. Es decir, cuando el entrenamiento es limitado, extender la cantidad de partidas no se traduce en una mejora significativa del aprendizaje ni en un refinamiento de las estrategias aprendidas.

## Explotabilidad real



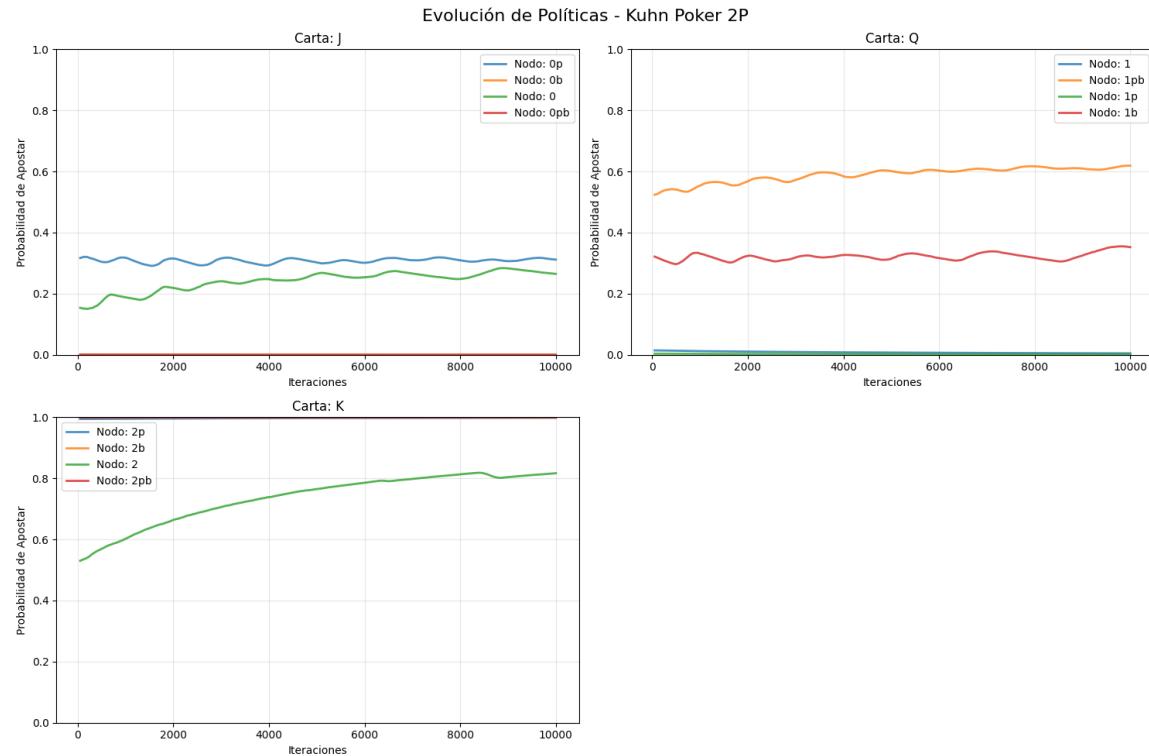
La explotabilidad muestra un perfil oscilante, con una ligera tendencia descendente. A diferencia del caso con 2000 partidas, aquí la curva es más detallada y presenta variaciones más suaves gracias a la mayor cantidad de datos.

A pesar de esto, la explotabilidad se mantiene por fuera de la zona de equilibrio (<5% del valor máximo), lo que indica que el agente aún puede ser explotado por oponentes mejor entrenados.

El hecho de que la curva no caiga significativamente reafirma que el bajo entrenamiento inicial no genera estrategias robustas.

#### 4.2.1.3 Entrenamiento medio (1000 iteraciones) y Corrida de estudio de 10.000 partidas

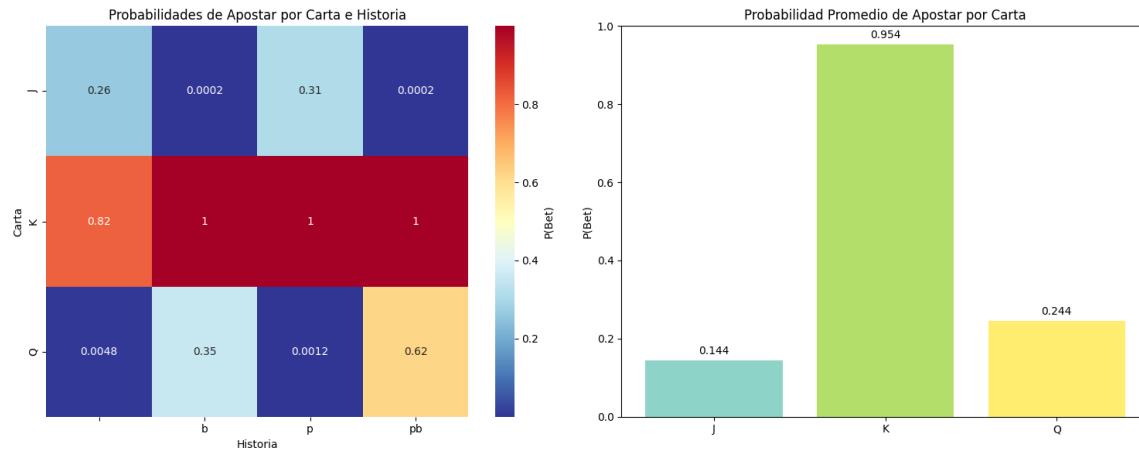
##### Evolucion de la politica:



Las políticas muestran una mayor definición respecto a las observadas en el primer bloque. Para la carta K, la probabilidad de apostar crece de forma progresiva y sostenida, alcanzando valores superiores al 80%, lo cual indica un alineamiento creciente con el equilibrio de Nash, donde esta carta representa una ventaja clara.

En cambio, las curvas asociadas a J y Q presentan un comportamiento más estable que en el bloque anterior, aunque todavía con leves oscilaciones. Esto sugiere que el agente comienza a estabilizar su política, pero sigue ajustando decisiones en escenarios donde las diferencias de valor esperado entre acciones son más sutiles.

### Heatmap de decisiones y Promedio de probabilidad de apostar por carta:

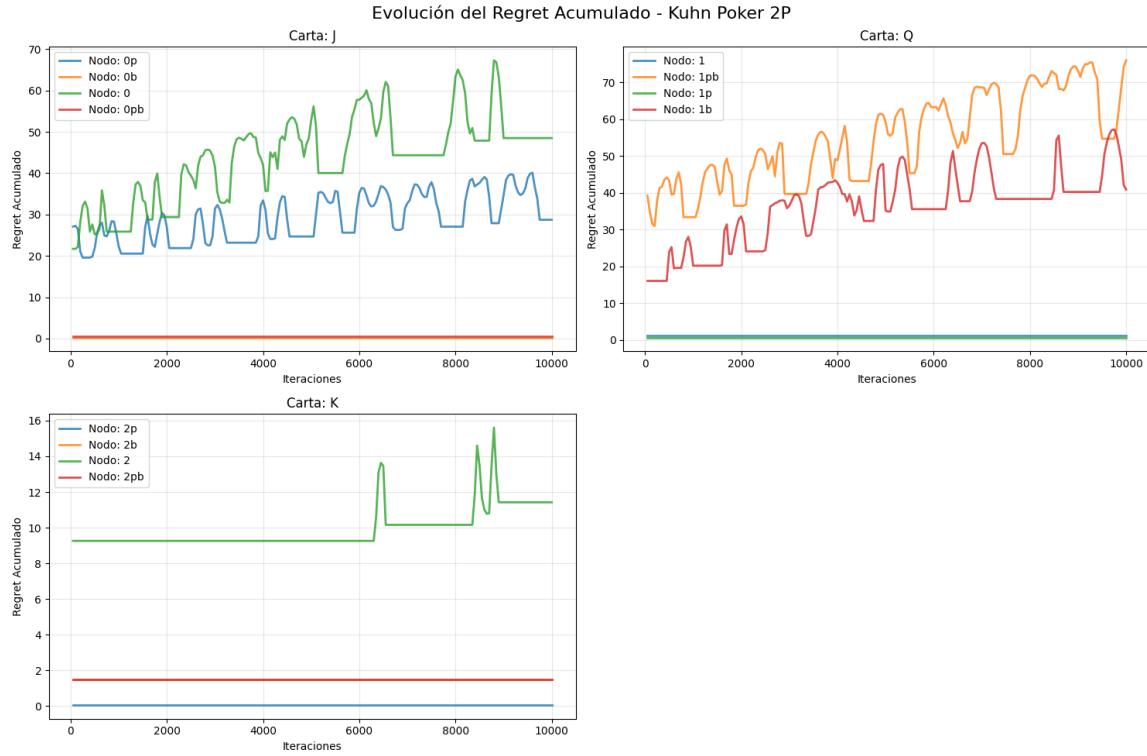


El heatmap refuerza la interpretación anterior: la carta K exhibe apuestas casi sistemáticas en todos los históricos, especialmente en aquellos con posibilidad de maximizar la ganancia, lo cual es consistente con una política racional. Por su parte, J y Q muestran patrones más heterogéneos. En particular, Q evidencia un mayor grado de apuesta en algunos históricos (como “pb” y “b”), aunque sin alcanzar niveles cercanos a la política óptima.

En cuanto al promedio de apuestas, se observa:

- K: 0.954; Muy cercano al valor teórico ideal.
- Q: 0.244; Comportamiento intermedio, con mezcla aún inestable.
- J: 0.144; Probabilidad de apuesta baja, aunque alineada con lo esperable para una carta débil.

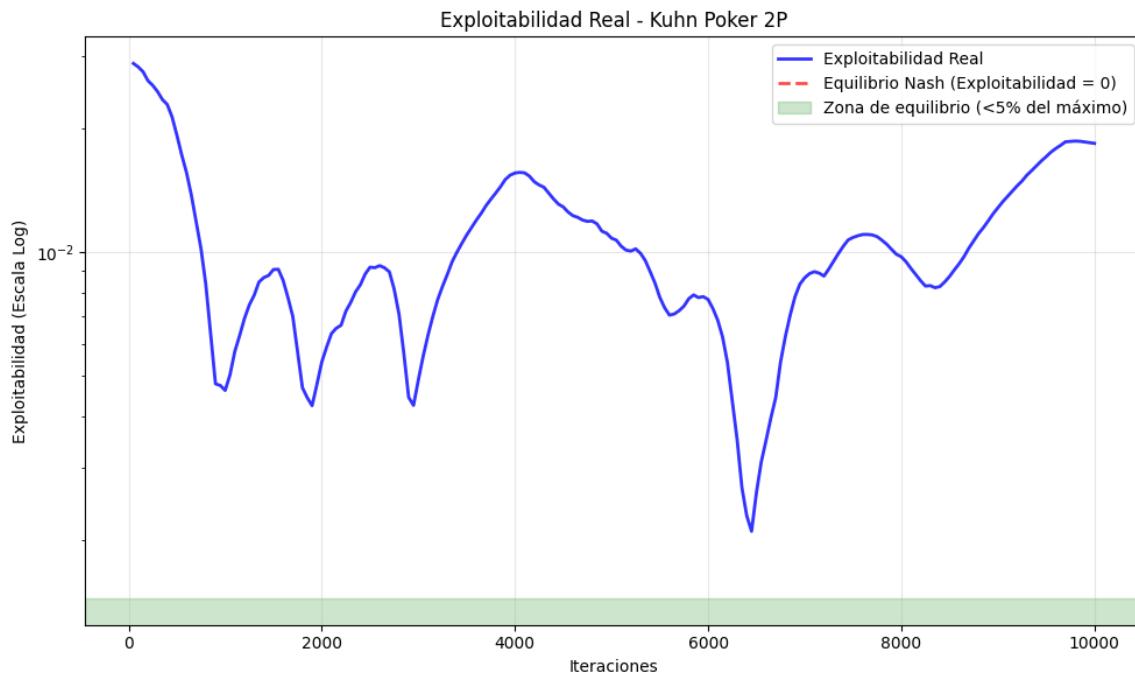
## Evolución del regret acumulado:



El comportamiento del regret acumulado mejora en relación con el bloque anterior. Las cartas **J** y **Q** aún presentan oscilaciones, especialmente en nodos con decisiones críticas (como “0p” o “1pb”), pero estas son menos abruptas y con tendencia a estabilizarse. Esto indica que el agente está comenzando a identificar con mayor precisión qué acciones son más rentables en cada nodo, aunque persiste cierta exploración.

Para la carta **K**, se mantiene un regret acumulado bajo y estable en todos los nodos, lo que sugiere una política consolidada para estas situaciones.

## Explotabilidad real

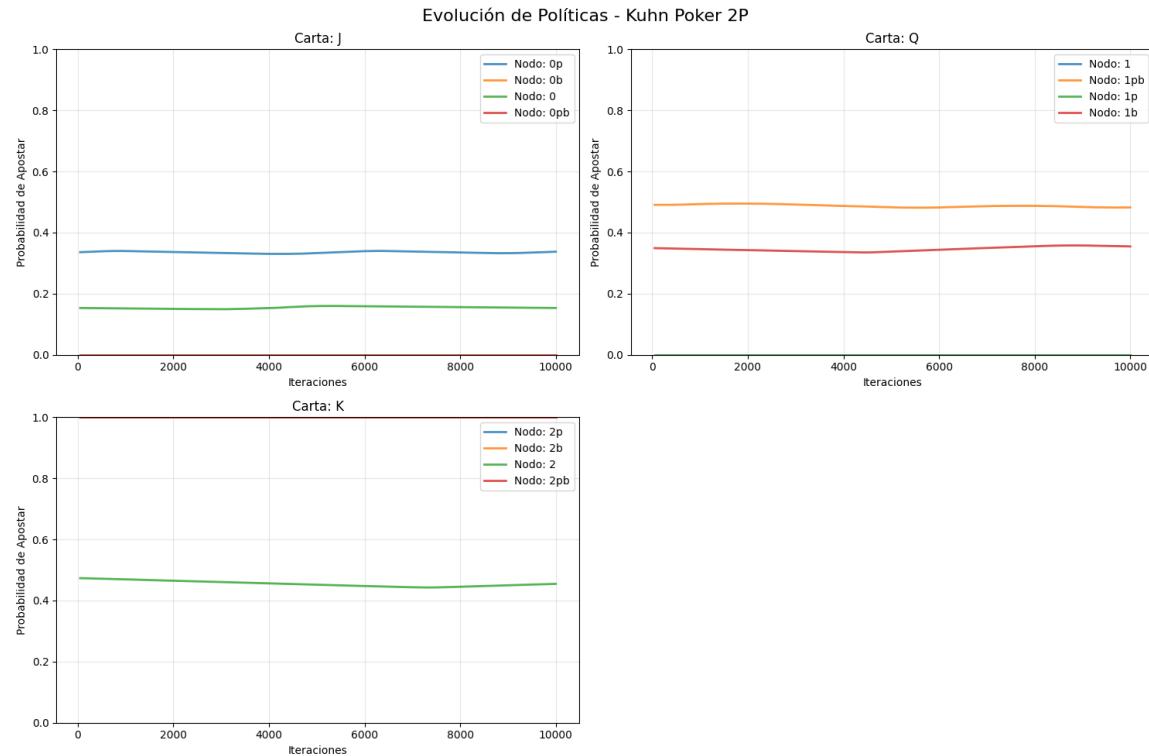


La explotabilidad real disminuye de forma marcada en las primeras iteraciones y luego presenta una dinámica ondulante dentro de un intervalo reducido. En comparación con el bloque anterior, se logra ingresar más frecuentemente en la zona de equilibrio (<5% del máximo), aunque aún no se alcanza una estabilidad sostenida.

Esto refleja un progreso efectivo en la calidad de la política, aunque con margen de mejora en la convergencia fina.

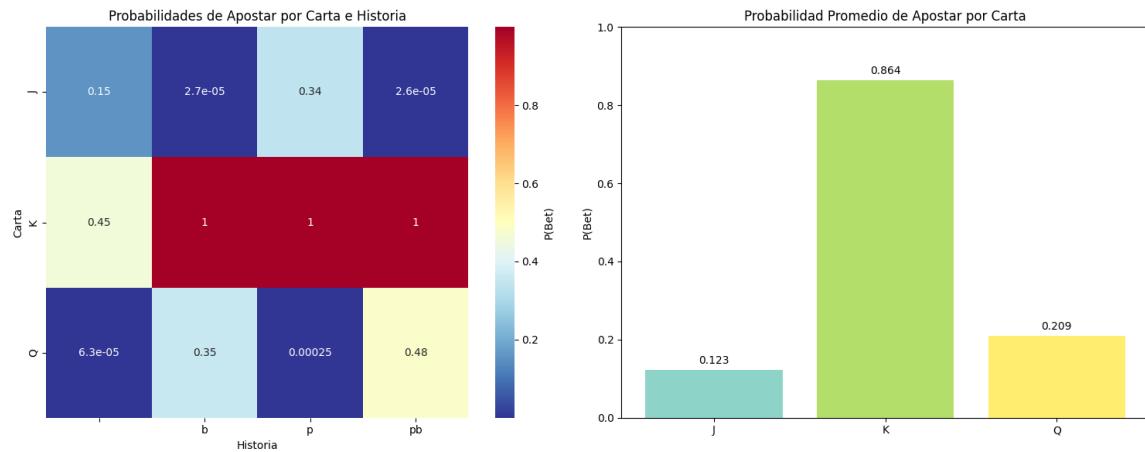
#### 4.2.1.4 Entrenamiento largo (100.000 iteraciones) y Corrida de estudio de 10.000 partidas

##### Evolucion de la politica:



Las curvas de evolución de política muestran una mayor estabilidad respecto a los experimentos previos, lo cual es esperable dado el nivel de entrenamiento alcanzado. La carta K mantiene una política fuertemente consistente, con probabilidades cercanas a 1 para apostar en todos los nodos, lo que refleja una clara convergencia hacia la estrategia dominante. En cambio, para J y Q se observan aún ligeras oscilaciones en algunos nodos, aunque menos pronunciadas que en las corridas con menor entrenamiento. Esto sugiere que el agente ha comenzado a consolidar sus decisiones, aunque ciertos nodos aún presentan ambigüedad estratégica que no ha sido completamente resuelta.

## Heatmap de decisiones y Promedio de probabilidad de apostar por carta:

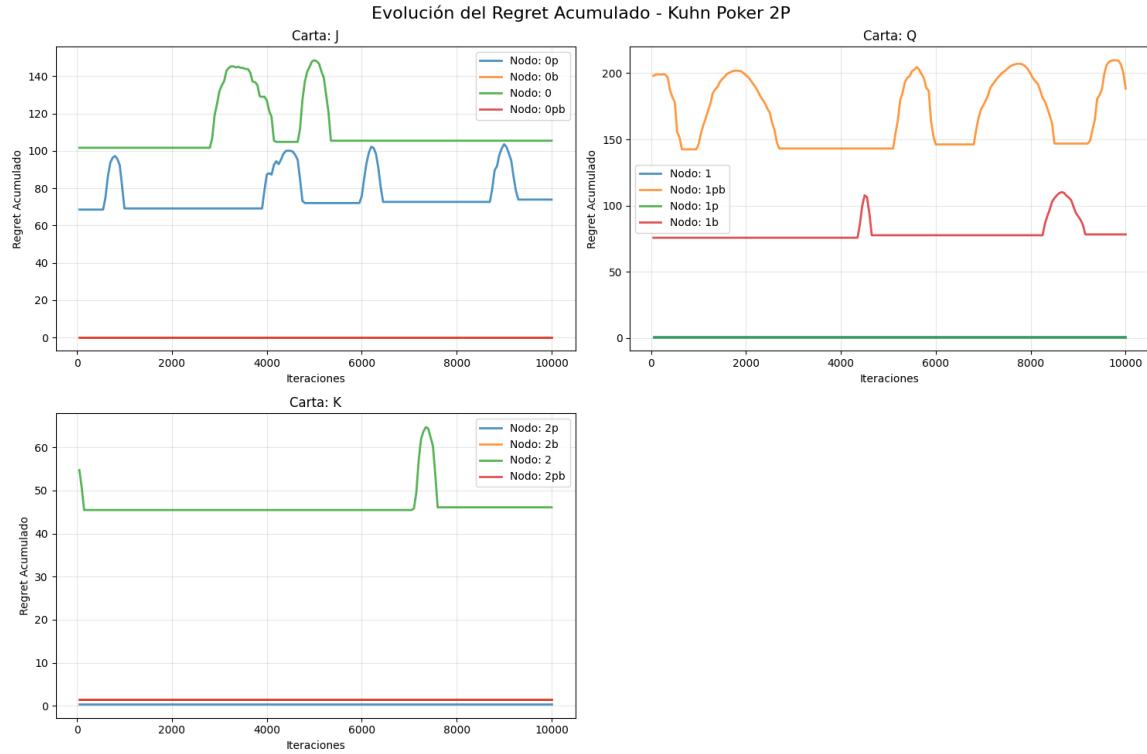


El mapa de calor muestra un patrón claro de decisión para la carta K, que apuesta con alta probabilidad en todos los contextos, evidenciando una política consolidada. En contraste, las decisiones con Q y J dependen más del historial de juego: por ejemplo, con Q tiende a apostar más en contextos donde ya ha habido una apuesta previa del oponente.

En los promedios globales, se observa:

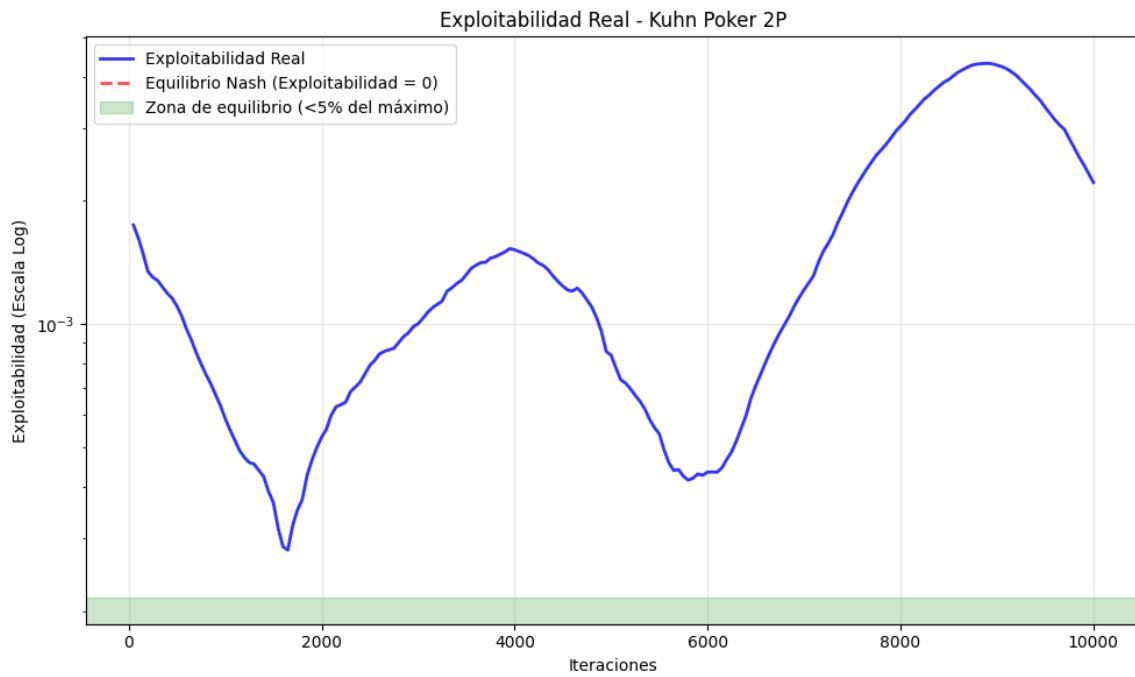
- K: 0.864; comportamiento altamente alineado con el equilibrio esperado.
- Q: 0.209; consistente con una estrategia mixta, aunque levemente por debajo del valor típico de equilibrio ( $\sim 1/3$ ).
- J: 0.123; menor propensión a apostar, como corresponde a una carta débil, aunque ligeramente inferior a los valores ideales.

## Evolución del regret acumulado:



Los gráficos de regret acumulado confirman el avance en el aprendizaje. Los nodos relacionados con K presentan regret prácticamente nulo en todas las etapas, lo cual es consistente con una política estabilizada y correcta. En los casos de J y Q, se identifica un patrón de oscilaciones marcadas en ciertos nodos (por ejemplo, 0p y 2 con J; 1pb y 1b con Q), lo cual sugiere que, aunque el entrenamiento ha reducido el error general, la exploración persiste en algunos contextos con decisiones más sensibles a la estrategia del oponente. No obstante, el comportamiento general refleja una tendencia hacia la disminución del regret en comparación con experimentos menos entrenados.

## Explotabilidad real



La explotabilidad sigue una trayectoria oscilante, alcanzando valores mínimos en el entorno de las 1.000–1.500 iteraciones, pero incrementándose posteriormente. A pesar de este aumento, los valores permanecen en un rango bajo en escala logarítmica, lo cual indica que el agente ha adquirido una política relativamente robusta frente a adversarios que buscan explotarlo. Este comportamiento sugiere que el entrenamiento permitió alcanzar regiones cercanas al equilibrio de Nash, aunque no logró una convergencia estable durante toda la corrida.

#### *4.2.1.5 Resumen*

A lo largo de los tres niveles de entrenamiento evaluados (corto, medio y largo), se observa una evolución progresiva del comportamiento del agente CFR, tanto en términos de regret acumulado como de políticas y explotabilidad.

- En el entrenamiento corto (10 iteraciones), el agente no logra estabilizar su política. El regret acumulado muestra variaciones pronunciadas, principalmente en los nodos asociados a las cartas J y Q, mientras que la carta K exhibe una política más estable desde el inicio. La alta variabilidad sugiere una fase de exploración activa sin una preferencia clara por estrategias dominantes.
- Durante el entrenamiento medio (1000 iteraciones), las políticas comienzan a mostrar cierta regularidad, aunque aún se evidencian oscilaciones marcadas en algunos nodos. La carta K continúa manteniendo una política consistente, mientras que J y Q presentan comportamientos menos convergentes. La explotabilidad disminuye respecto al bloque anterior, pero con tendencia osculatoria, lo cual indica avances parciales en el aprendizaje.
- En el entrenamiento largo (100.000 iteraciones), se consolida el comportamiento estratégico del agente. La política asociada a la carta K es sistemática y estable en todos los contextos, reflejando una correcta internalización de la estrategia óptima. Si bien algunos nodos vinculados a las cartas J y Q mantienen cierto nivel de regret acumulado, las decisiones son notablemente más coherentes. La explotabilidad alcanza niveles bajos, aunque persiste una leve inestabilidad, posiblemente asociada a la sensibilidad del entorno a cambios pequeños en las políticas.

Conclusión general:

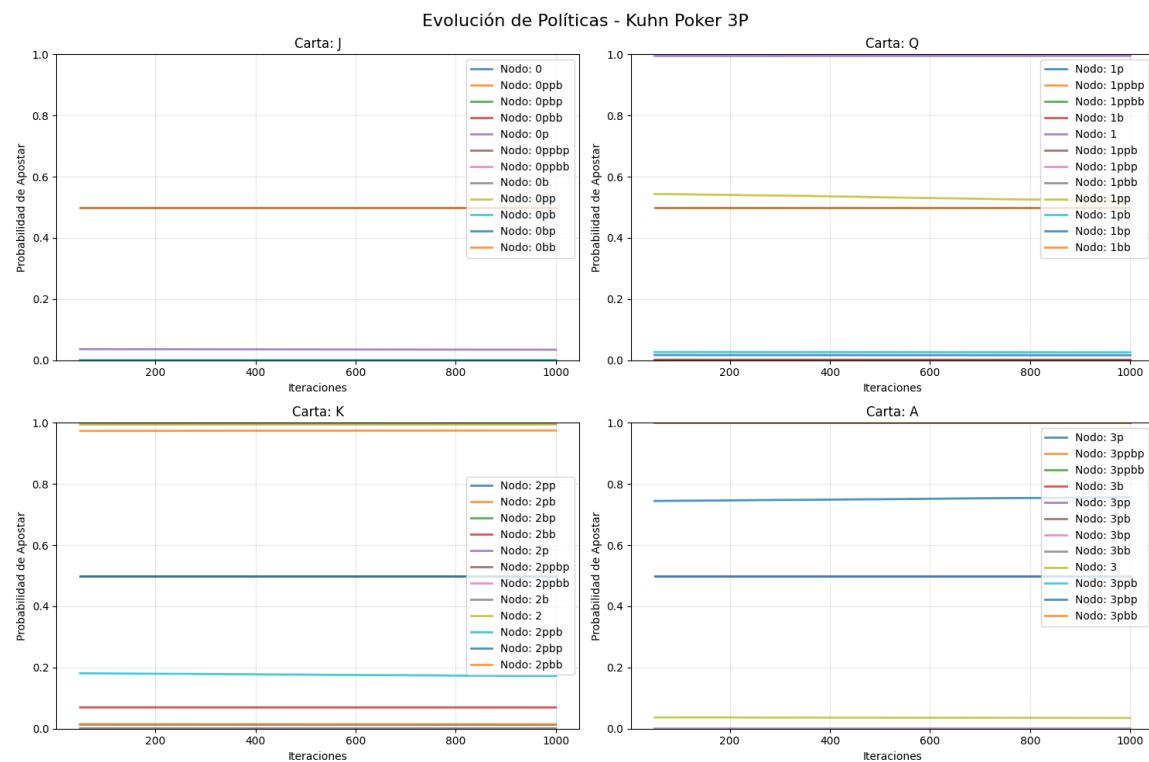
El agente CFR muestra una clara progresión en su capacidad de aprendizaje a medida que se incrementan las iteraciones. Sin embargo, la convergencia plena al equilibrio de Nash no se alcanza de forma sostenid. Esto sugiere que, incluso con entrenamiento prolongado, ciertas decisiones requieren un refinamiento adicional para estabilizarse completamente.

## 4.2.2 Khun poker 3 jugadores

En este entorno, al igual que en el anterior, se analiza el desempeño de un agente basado en *Counterfactual Regret Minimization* (CFR), evaluando tanto el efecto del número de iteraciones de entrenamiento como la influencia de la duración de las partidas en su comportamiento estratégico.

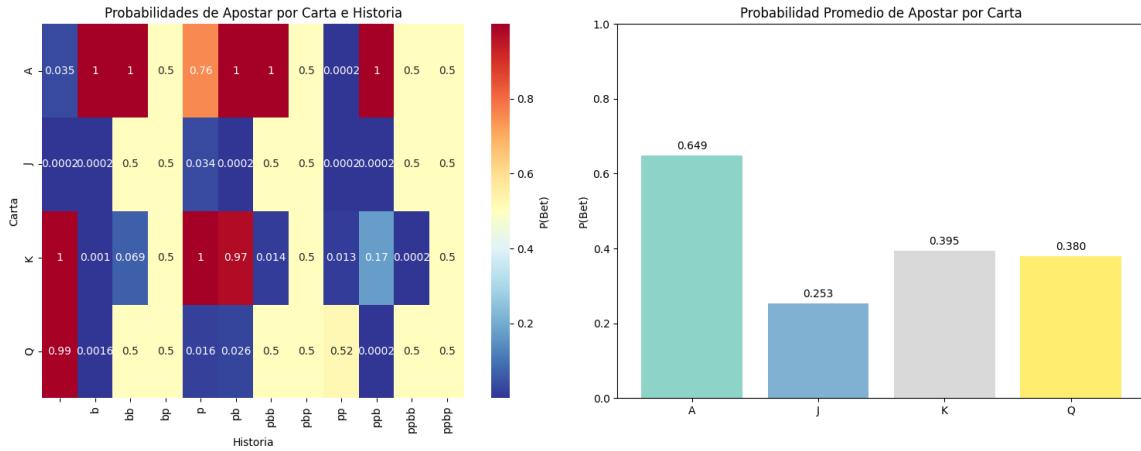
### 4.2.2.1 Entrenamiento corto (10 iteraciones) y Corrida de estudio de 1.000 partidas

Evolucion de la politica:



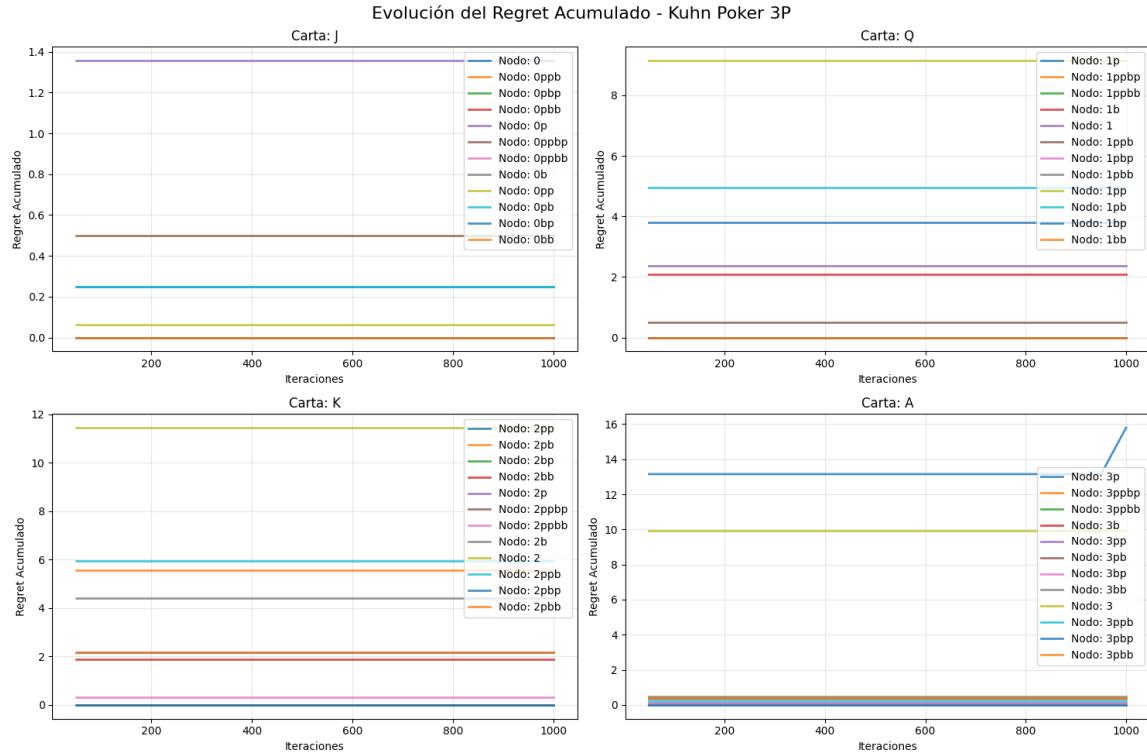
Las políticas aprendidas se mantienen constantes a lo largo de las 10 iteraciones. Esto es esperable dado el bajo número de iteraciones, que no resulta suficiente para permitir una adaptación significativa de las estrategias. Las probabilidades de apostar se distribuyen de forma fija por historia y carta, sin mostrar señales de convergencia hacia patrones estratégicos.

## Heatmap de decisiones y Promedio de probabilidad de apostar por carta:



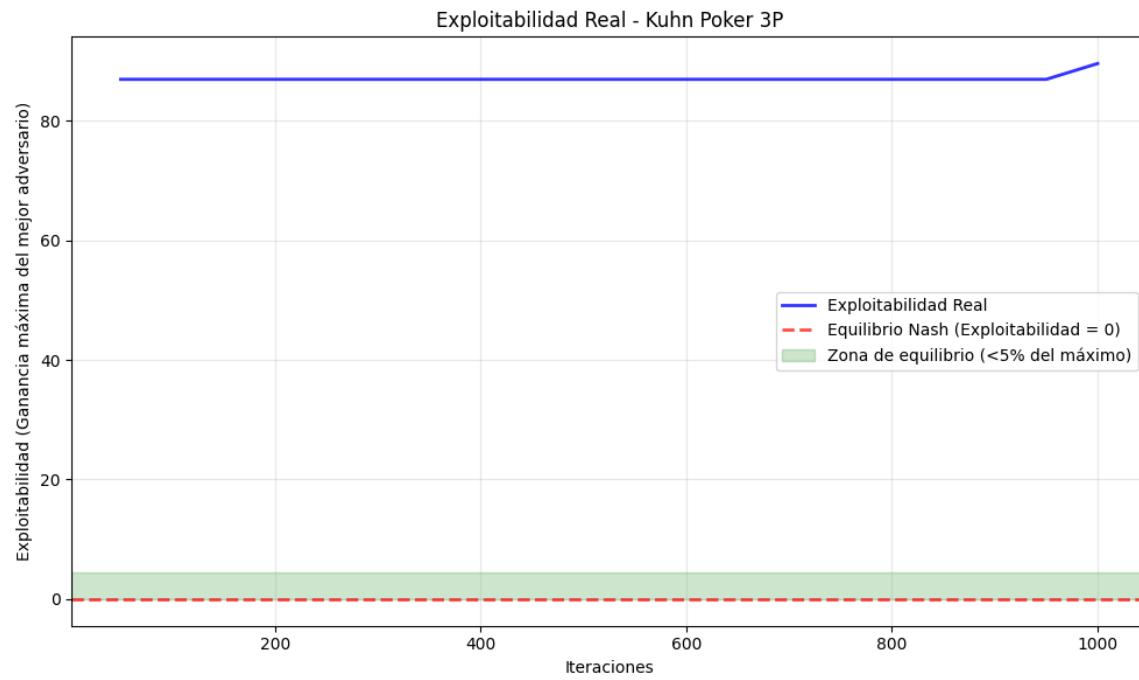
El mapa de calor muestra una alta variabilidad en las probabilidades de apuesta entre combinaciones de cartas e historias, muchas de ellas rondando el 0.5, lo cual sugiere comportamientos aleatorios. Algunas decisiones muestran probabilidades extremas (cercaas a 0 o 1) sin una justificación estratégica clara, lo que refuerza la conclusión de que el agente aún no ha logrado aprender patrones relevantes.

## Evolución del regret acumulado:



El regret acumulado permanece bajo en casi todos los nodos, pero esto no debe interpretarse como una señal de convergencia. En realidad, se debe al escaso entrenamiento, lo que impide que el agente acumule suficiente experiencia como para evaluar y corregir decisiones subóptimas. El comportamiento plano en todos los nodos confirma la falta de aprendizaje efectivo.

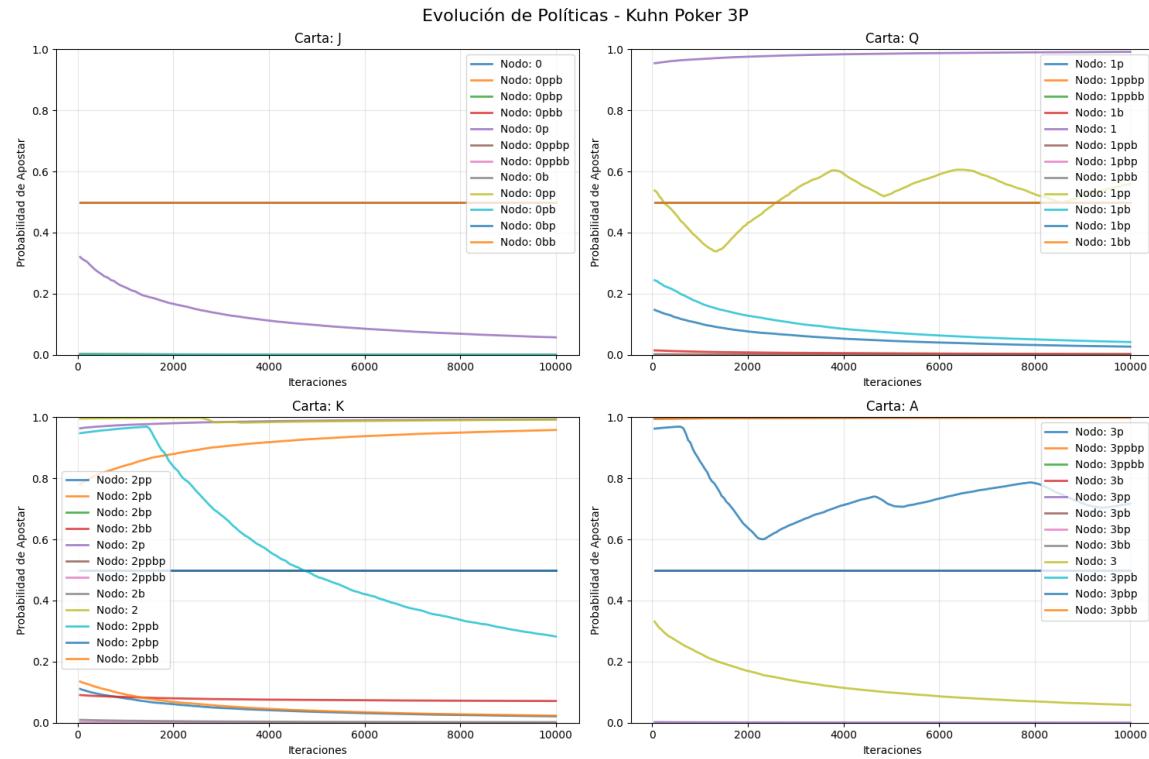
## Explotabilidad real



La explotabilidad del agente es alta y constante, lo que indica que sus estrategias son fácilmente explotables por un adversario óptimo. Además, se encuentra muy por fuera de la zona de equilibrio (zona verde), lo cual refuerza que no ha habido progreso en dirección a una estrategia cercana a Nash.

#### 4.2.2.2 Entrenamiento corto (10 iteraciones) y Corrida de estudio de 10.000 partidas

##### Evolucion de la politica:

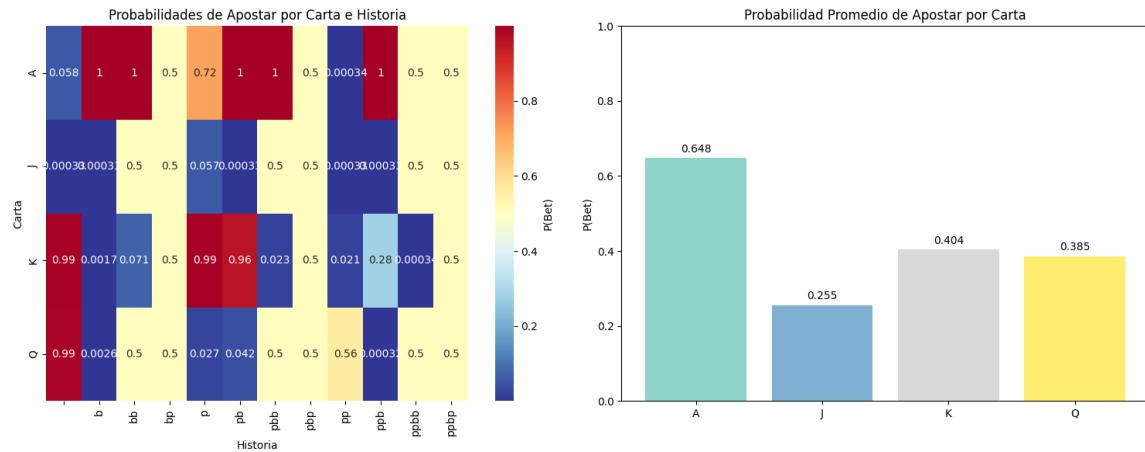


La evolución de las probabilidades de apostar muestra cierto dinamismo en este escenario, a diferencia del torneo corto:

- Carta J: Se observan políticas que tienden a estabilizarse en valores bajos. Esto sugiere que el agente identifica que J es una carta débil y ajusta su comportamiento de forma conservadora.
- Carta Q: Presenta un comportamiento oscilatorio en algunos nodos, en particular 1pb y 1ppb, sin una convergencia clara. Esto indica que el agente no logra consolidar una política estable para esta carta intermedia.
- Carta K: Se aprecia un patrón más definido, con ciertos nodos alcanzando valores cercanos a 1 o a 0. En particular, 2pb y 2pbb muestran una tendencia creciente hacia la apuesta, reflejando una confianza creciente del agente al contar con esta carta.

- Carta A: Las decisiones asociadas a A fluctúan entre valores altos y medios. Aunque algunos nodos alcanzan probabilidad de apuesta igual a 1, otros como 3ppb muestran una caída inicial seguida de oscilaciones, lo que indica un intento de adaptación sin consolidación.

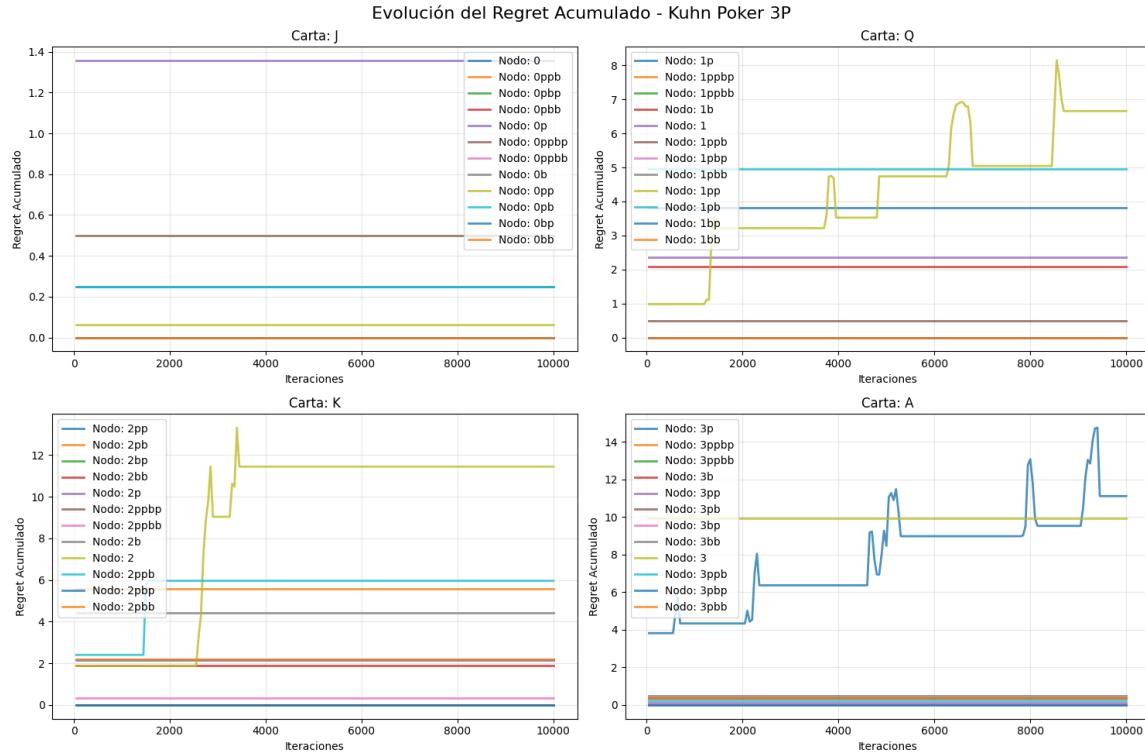
#### Heatmap de decisiones y Promedio de probabilidad de apostar por carta:



El heatmap evidencia que para las cartas fuertes (K y A), existen varios nodos con probabilidad de apuesta cercana a 1, en especial en las primeras rondas (b, p). Sin embargo, para J, las decisiones son considerablemente más conservadoras, con valores cercanos a cero en casi todos los nodos.

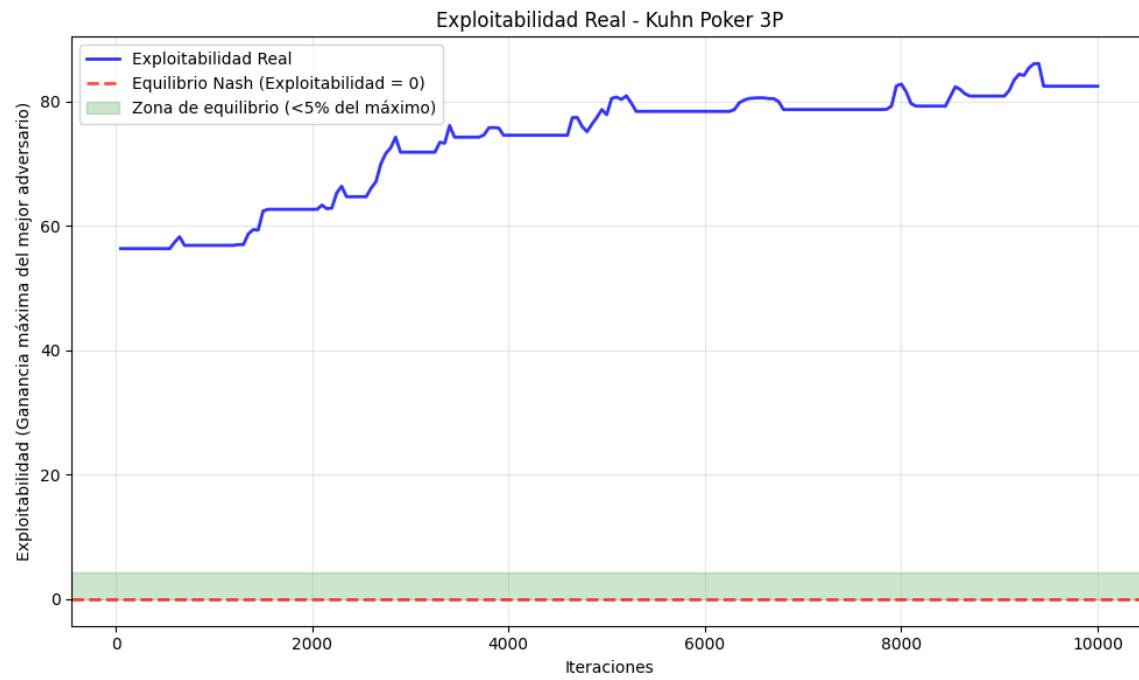
El promedio de apuesta por carta refuerza esta observación: la carta A mantiene un promedio de 0.648, seguida por K (0.404) y Q (0.385), mientras que J queda rezagada con 0.255. Esto sugiere que el agente aplica una política general de agresividad creciente con cartas más fuertes, aunque aún con incertidumbre en cartas intermedias como Q.

## Evolución del regret acumulado:



- Carta J: El regret se mantiene estable, reflejando que las decisiones conservadoras del agente no son significativamente penalizadas, aunque tampoco ajustadas.
- Carta Q: Se observan picos de regret crecientes en nodos como 1ppb, lo cual indica que las decisiones en estas historias generan oportunidades de mejora que el agente no logra capitalizar.
- Carta K: Algunos nodos como 2pb y 2pbb presentan saltos pronunciados en el regret, lo que sugiere errores estratégicos persistentes o falta de consistencia en la toma de decisiones frente a esta carta fuerte.
- Carta A: El nodo 3p presenta oscilaciones crecientes en el regret, señal de una estrategia poco refinada pese a tratarse de la carta más fuerte. Otros nodos mantienen valores bajos, pero sin ajustes visibles.

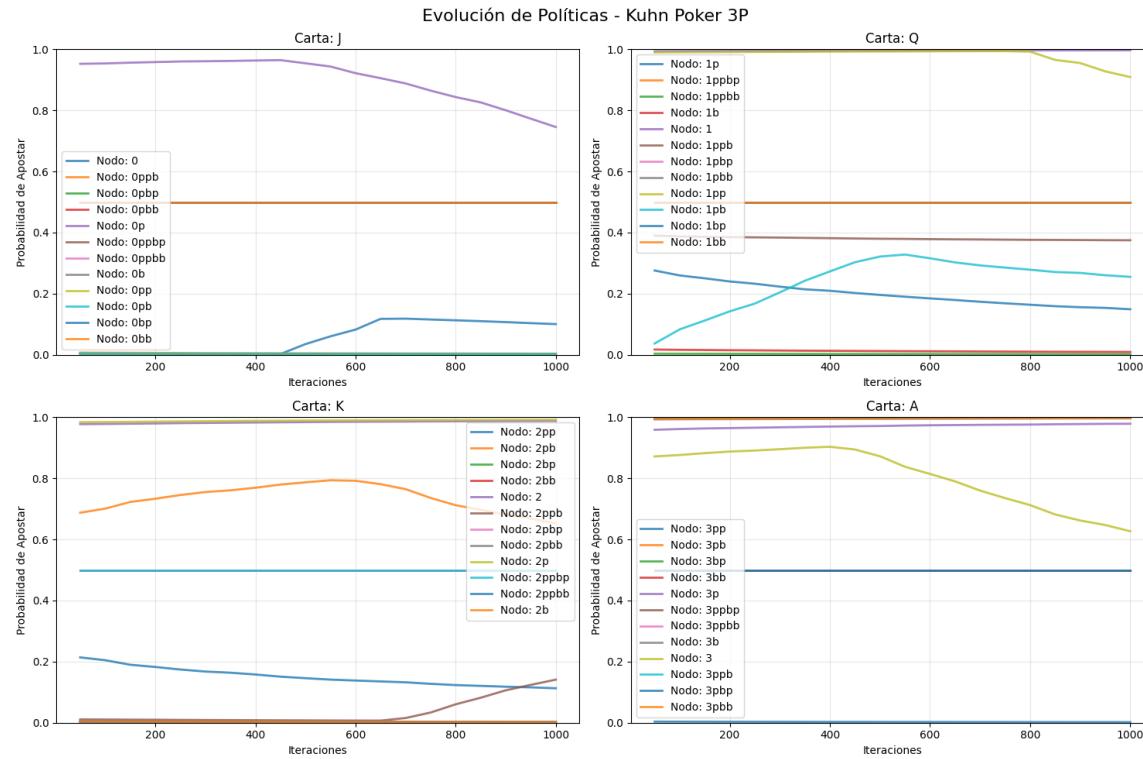
## Explotabilidad real



La explotabilidad presenta una tendencia creciente, pasando de valores en torno a 56 hasta estabilizarse en más de 80. Este comportamiento indica que el aumento en el número de partidas no mejora la calidad de la estrategia, sino que puede incluso amplificar errores al no estar respaldado por un entrenamiento suficiente.

#### 4.2.2.3 Entrenamiento media (1.000 iteraciones) y Corrida de estudio de 1.000 partidas

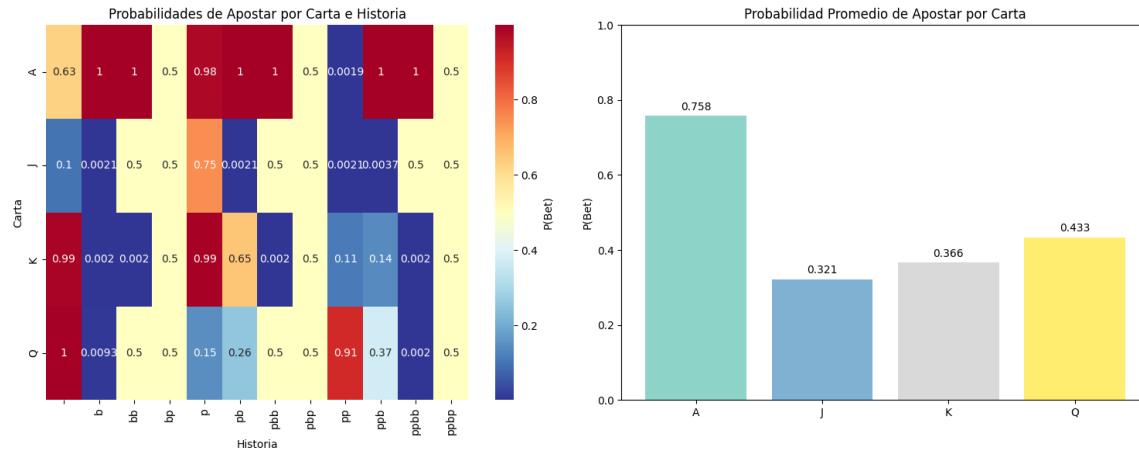
##### Evolucion de la politica:



- Carta J: Las probabilidades de apostar en la mayoría de los nodos tienden a disminuir, aunque sin alcanzar valores estables. Se observa una caída marcada en algunos nodos como 0ppbp, lo cual sugiere ajustes tempranos del agente, aún sin convergencia.
- Carta Q: Las curvas muestran una fase activa de exploración. Algunos nodos como 1bp y 1p varían significativamente a lo largo del entrenamiento, lo que indica que el agente aún está evaluando distintas respuestas estratégicas para esta carta.
- Carta K: La política muestra señales de mayor estabilidad. Las acciones tienden a consolidarse, especialmente en nodos con historia corta como 2p y 2pb, aunque otros como 2b aún presentan oscilaciones suaves.

- Carta A: El comportamiento tiende a estabilizarse más rápido, con altas probabilidades de apuesta en nodos claves (3p, 3pbpb, etc.), lo que es coherente con una política agresiva al tener la mejor carta.

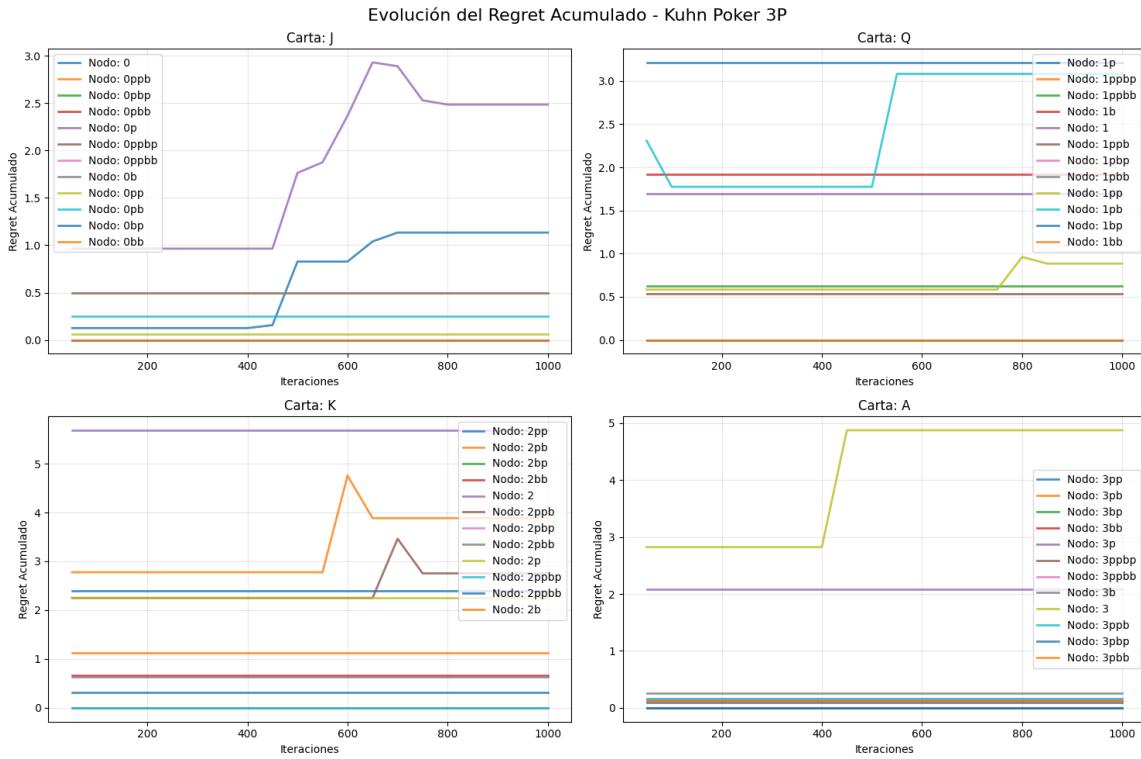
### Heatmap de decisiones y Promedio de probabilidad de apostar por carta:



El mapa de calor indica una política todavía inmadura: muchas celdas muestran probabilidades cercanas a 0.5, reflejando decisiones aleatorias o inestables. Solo en los casos extremos (A en posiciones de ventaja o J en desventaja), la política parece más decidida.

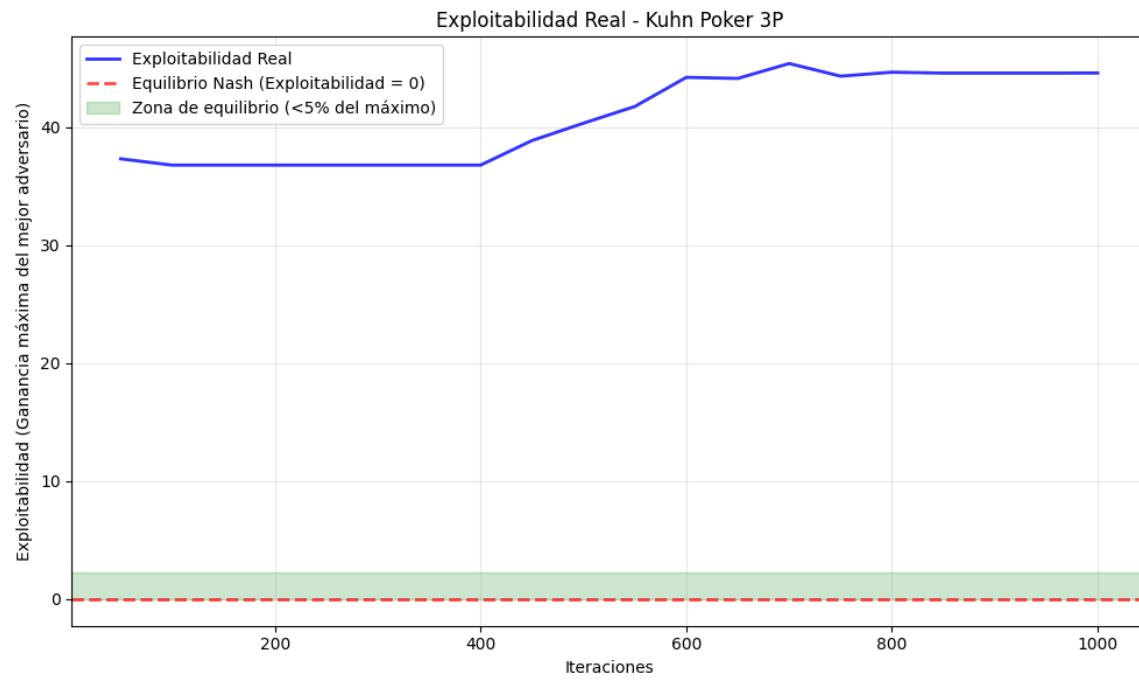
La carta A presenta una alta propensión a apostar (0.758), como se espera en una estrategia racional. Las demás cartas tienen valores intermedios (Q: 0.433, K: 0.366, J: 0.321), sin una diferenciación estratégica marcada, lo cual evidencia un nivel de aprendizaje todavía bajo.

## Evolución del regret acumulado:



- Carta J: Se observa un incremento relevante de regret en algunos nodos (0ppbp, 0), lo que indica que el agente aún no encuentra una política efectiva para esos estados.
- Carta Q: Algunos nodos como 1p y 1bp muestran una acumulación significativa de regret, lo cual sugiere dudas del agente sobre la mejor acción posible en esos contextos.
- Carta K: Regret acumulado en ciertos nodos como 2b y 2ppbp, reflejando dificultades para establecer acciones óptimas frente a historias específicas.
- Carta A: Aunque algunos nodos (3bp) presentan acumulaciones altas, la mayoría mantiene regret bajo, consistente con una política relativamente más estable.

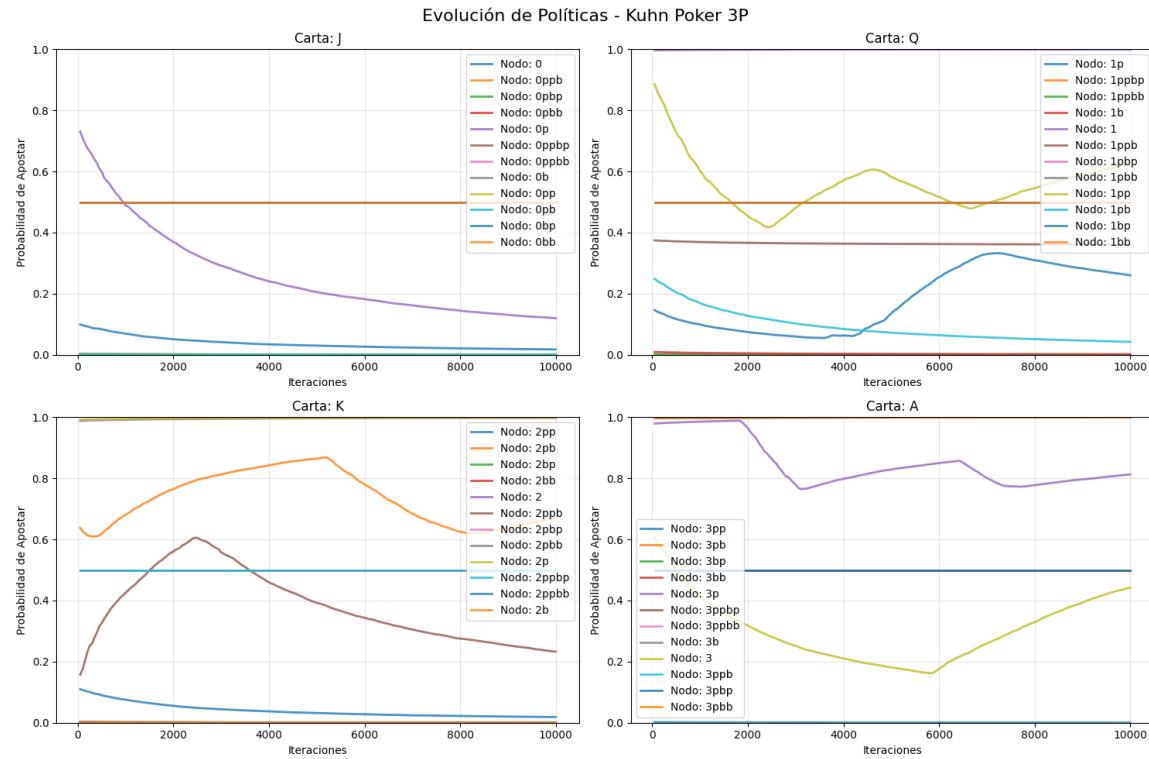
## Explotabilidad real



El nivel de explotabilidad ronda valores altos (~45), indicando que el agente todavía es predecible y vulnerable. Aunque hay una leve mejora frente al inicio, el desempeño dista mucho de un equilibrio.

#### 4.2.2.4 Entrenamiento medio (1.000 iteraciones) y Corrida de estudio de 10.000 partidas

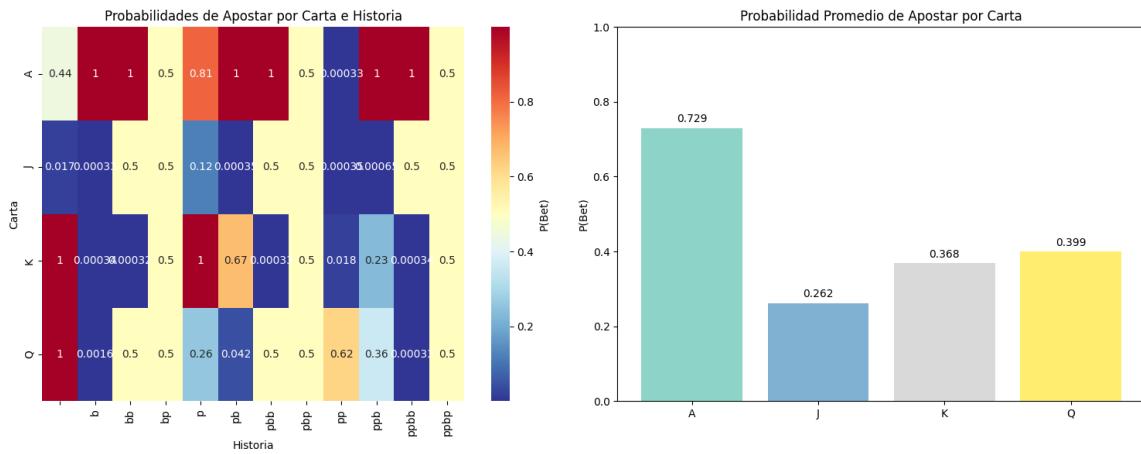
##### Evolucion de la politica:



- Carta J: La política muestra un ajuste progresivo, con una clara disminución en la probabilidad de apostar en los nodos más relevantes, como 0ppbb. Esto sugiere una tendencia hacia una estrategia más conservadora en contextos donde apostar no es rentable con esta carta débil.
- Carta Q: La evolución es menos estable, con curvas que oscilan antes de encontrar un nivel intermedio. Esta inestabilidad indica que el agente aún está explorando decisiones ante situaciones más complejas donde Q puede ser una carta ambigua (ni débil ni fuerte).
- Carta K: Se evidencia una política activa, particularmente en 2b, que primero incrementa fuertemente su tendencia a apostar y luego retrocede. Este comportamiento sugiere un refinamiento adaptativo en función de los resultados durante el torneo, aunque todavía no se estabiliza.

- Carta A: La carta más fuerte mantiene un comportamiento más estable, con alta propensión a apostar en la mayoría de los nodos. La política en 3ppbb, aunque presenta cierta fluctuación, refuerza esta línea de acción agresiva coherente con la fuerza de la mano.

#### Heatmap de decisiones y Promedio de probabilidad de apostar por carta:

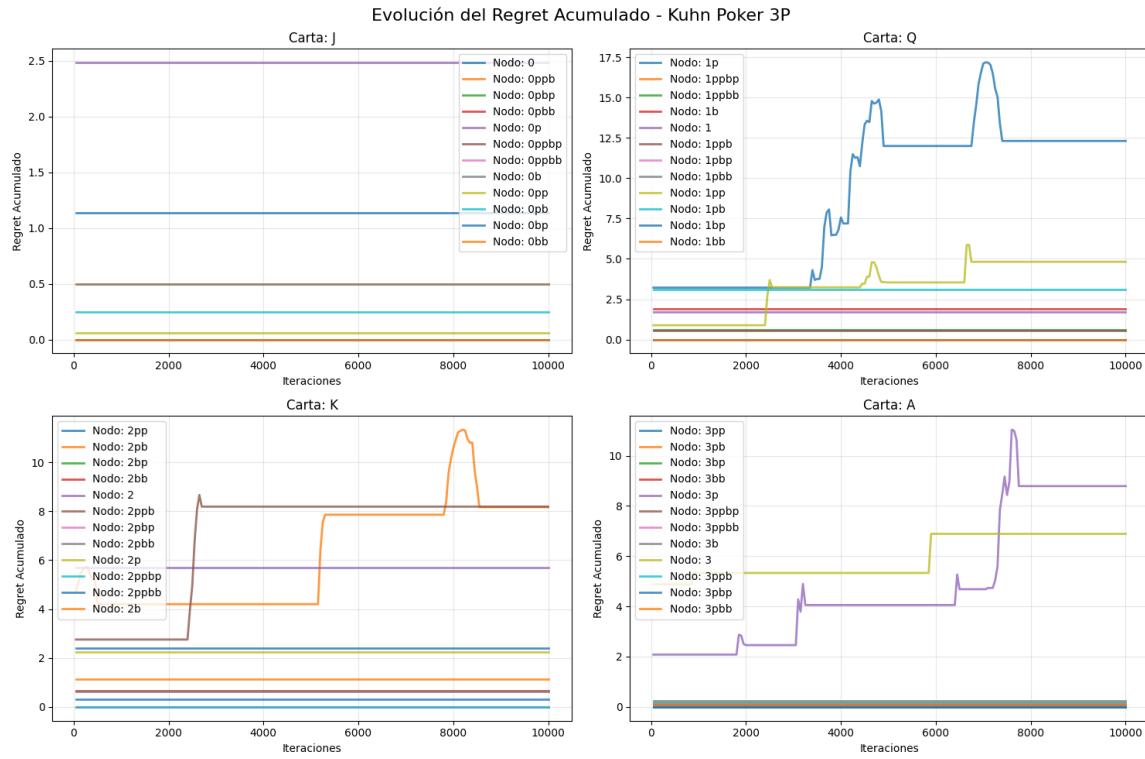


El heatmap confirma una tendencia general a apostar de forma agresiva con A, y mucho más selectiva con J.

En K y Q, se observan zonas de incertidumbre con probabilidades moderadas en varias historias, lo que refleja ambigüedad en la política aprendida.

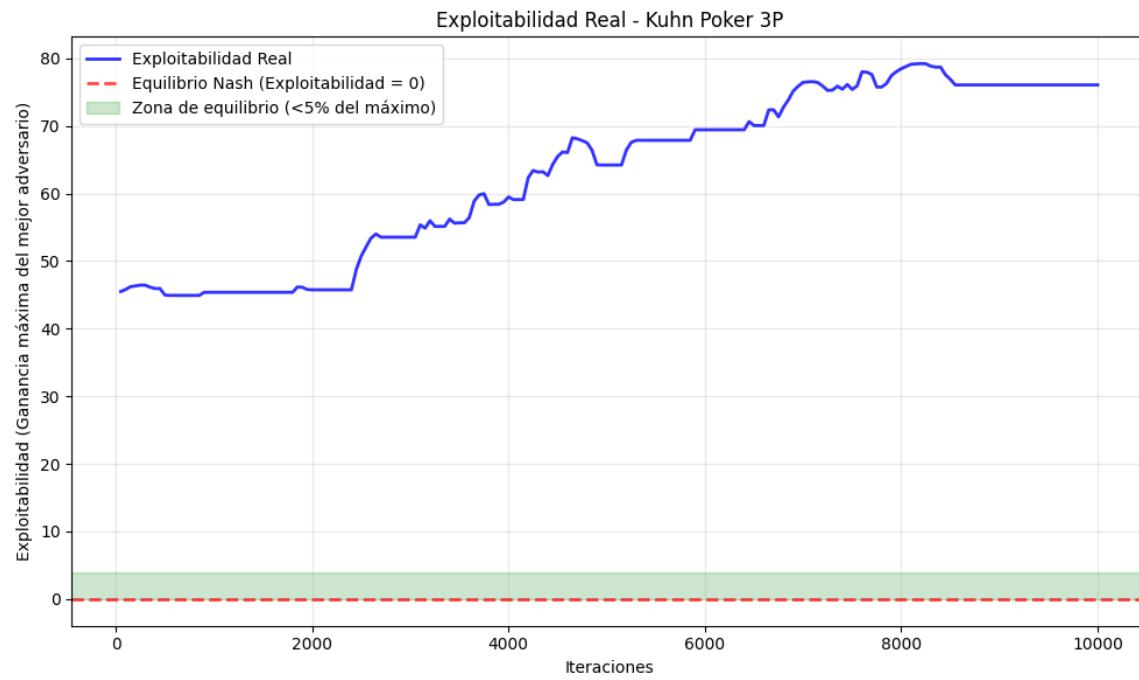
El promedio por carta refuerza esto: A mantiene la mayor propensión a apostar (0.729), mientras que J es la más conservadora (0.262). Q (0.399) y K (0.368) se encuentran en zonas intermedias, lo que concuerda con su comportamiento táctico más complejo.

## Evolución del regret acumulado:



- Carta J: El regret acumulado se mantiene estable, lo cual indica que el agente ha dejado de explorar decisiones alternativas para esta carta, reforzando su política conservadora.
- Carta Q: Se destacan fuertes aumentos en nodos como 1p, lo que denota conflicto entre alternativas estratégicas, con un alto grado de incertidumbre aún presente.
- Carta K: El nodo 2b presenta picos de regret importantes, lo cual refleja que esta acción fue evaluada como subóptima frente a alternativas, mostrando una política aún en revisión.
- Carta A: Se evidencian aumentos marcados de regret en nodos como 3ppbb, lo que sugiere que incluso con la mejor carta, no todas las líneas agresivas resultan consistentemente beneficiosas.

## Explotabilidad real

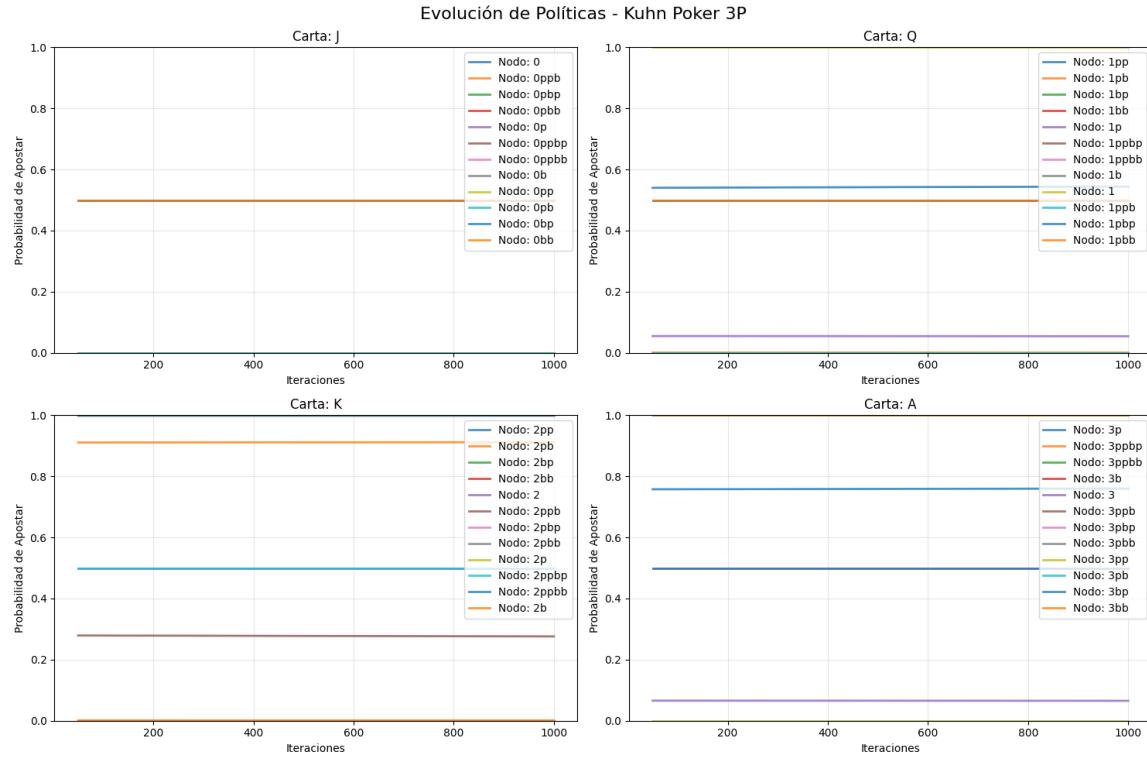


La explotabilidad muestra una tendencia creciente, alcanzando niveles cercanos a 80. Esto revela que, a pesar del mayor número de partidas en el torneo, el agente aún dista mucho de un comportamiento equilibrado.

Esta evolución confirma que las 1000 iteraciones de entrenamiento son insuficientes para generar políticas robustas, independientemente del largo del torneo utilizado para la evaluación.

#### 4.2.2.5 Entrenamiento largo (100.000 iteraciones) y Corrida de estudio de 1.000 partidas

##### Evolucion de la politica:

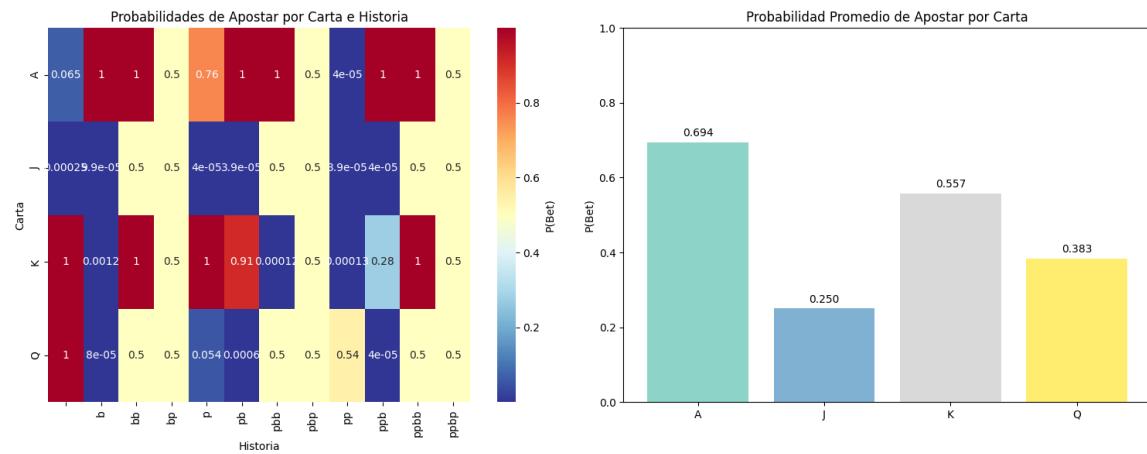


- Carta J: Las probabilidades de apuesta asociadas a esta carta son prácticamente nulas en todos los nodos. Esto sugiere que el agente ha aprendido a jugar de forma pasiva con esta mano débil, evitando generar pérdidas innecesarias. No se observan fluctuaciones: la política converge temprano y se mantiene inalterada.
- Carta Q: Las políticas son estables desde el inicio del torneo. La probabilidad de apostar se mantiene intermedia o baja según el nodo. No hay evolución a lo largo del tiempo, lo cual es coherente con un agente ya entrenado que ejecuta una política fija desde el comienzo. Esto contrasta con el comportamiento más dinámico observado en bloques anteriores.
- Carta K: Se mantiene una probabilidad de apuesta alta ( $\approx 0.91$ ) en algunos nodos, con otras decisiones distribuidas entre niveles medios y bajos según el historial de acciones. El comportamiento es estable, indicando que el agente

ha consolidado sus preferencias estratégicas para esta carta. No hay exploración activa durante el torneo.

- Carta A: La estrategia es fuertemente agresiva. En múltiples nodos se apuesta con probabilidad 1, sin variaciones durante el torneo. Esto es consistente con una política óptima para una mano dominante. Al igual que en los otros casos, no se observa ninguna evolución.

### Heatmap de decisiones y Promedio de probabilidad de apostar por carta:

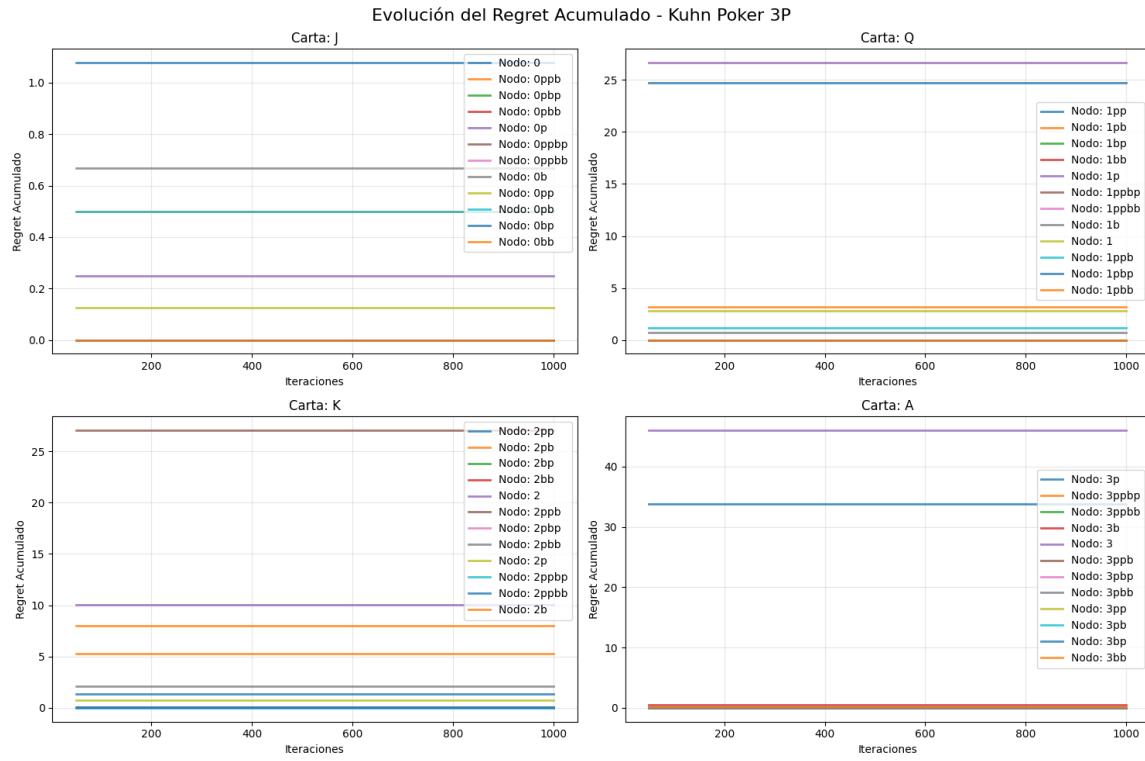


El heatmap muestra que las cartas A y K presentan alta probabilidad de apostar en múltiples historias, lo que es esperable dada su fortaleza.

Las cartas Q y J, en cambio, presentan comportamientos más cautelosos.

El gráfico de barras confirma esta tendencia. Las cifras muestran un orden jerárquico de agresividad coherente con la fuerza relativa de las cartas.

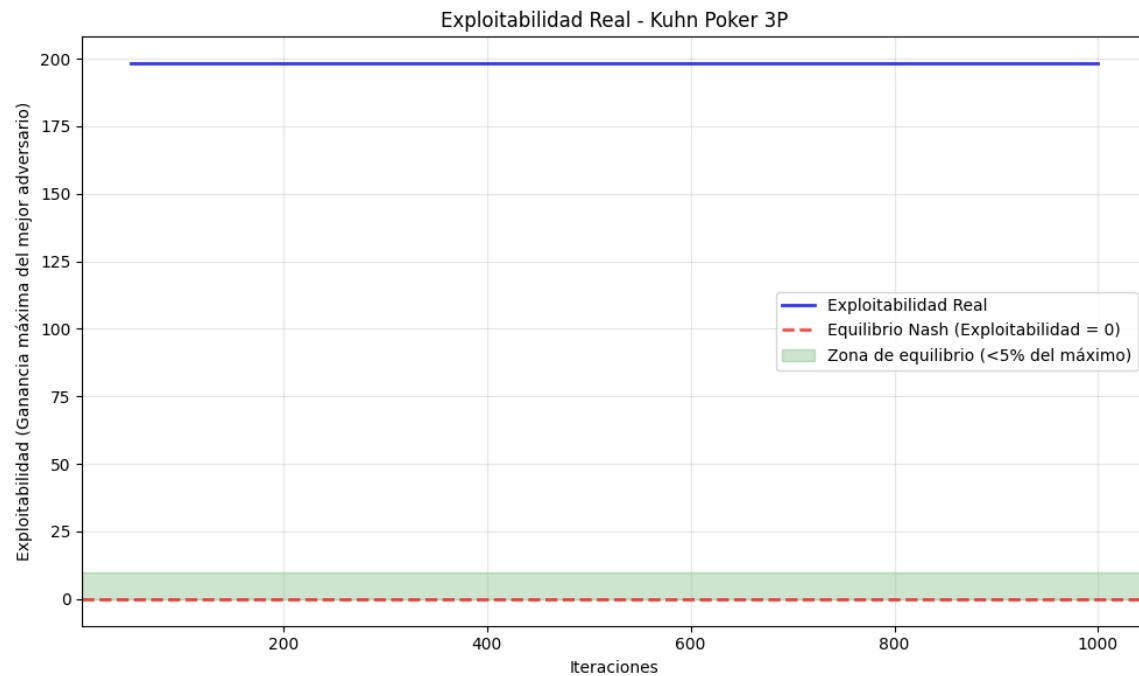
## Evolución del regret acumulado:



Los valores de regret son estáticos y están determinados desde el inicio del torneo. Esto sugiere que el entrenamiento previo fue suficientemente largo como para que el agente ya no esté realizando ajustes en tiempo de juego.

Algunos nodos específicos presentan valores altos (por ejemplo, en cartas K y A), pero estos se mantienen constantes, reflejando que no hay más actualización del regret. Es posible que estos nodos tengan menor frecuencia de visita o que contengan acciones subóptimas difíciles de corregir si no se observan con suficiente frecuencia.

## Explotabilidad real

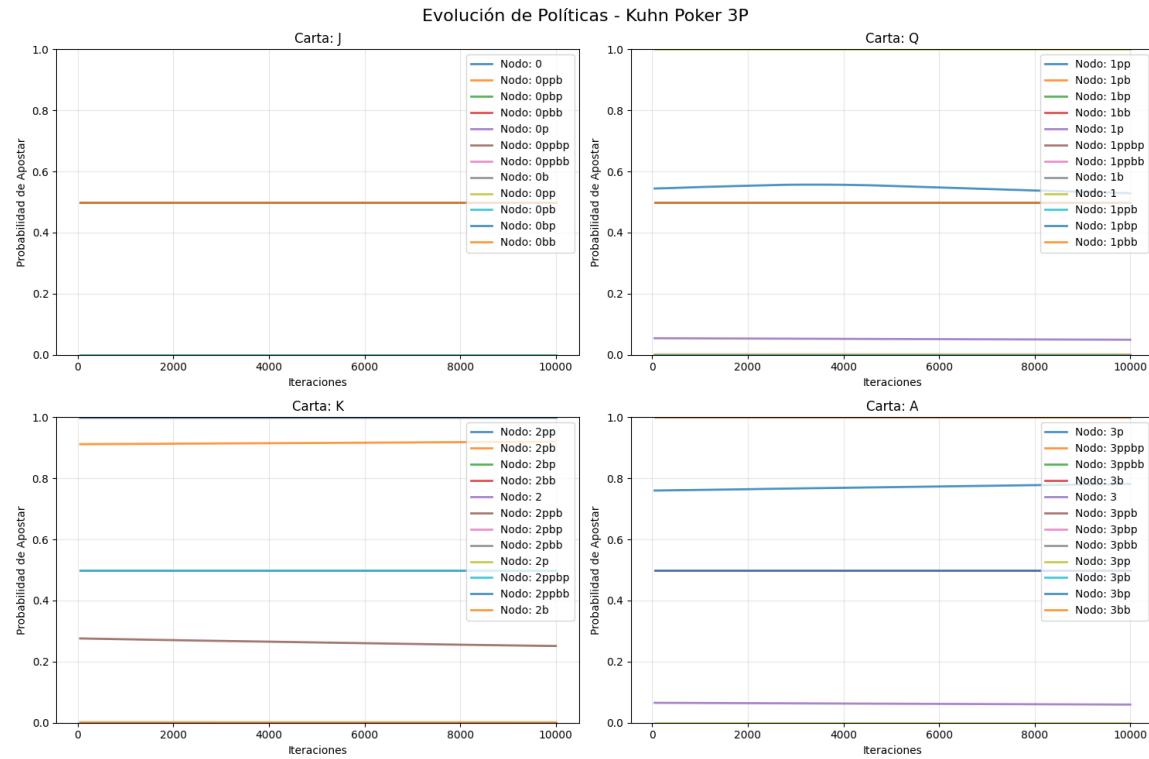


La explotabilidad se mantiene constante y alta (~200) durante todo el torneo. Esto indica que el agente, si bien ejecuta una política estable, no es robusto frente a un oponente perfecto.

La política resultante está lejos del equilibrio de Nash, lo que sugiere que el entrenamiento, aunque largo, no garantizó la convergencia teórica esperada. La falta de ajuste durante el torneo evidencia que el agente ya no está aprendiendo.

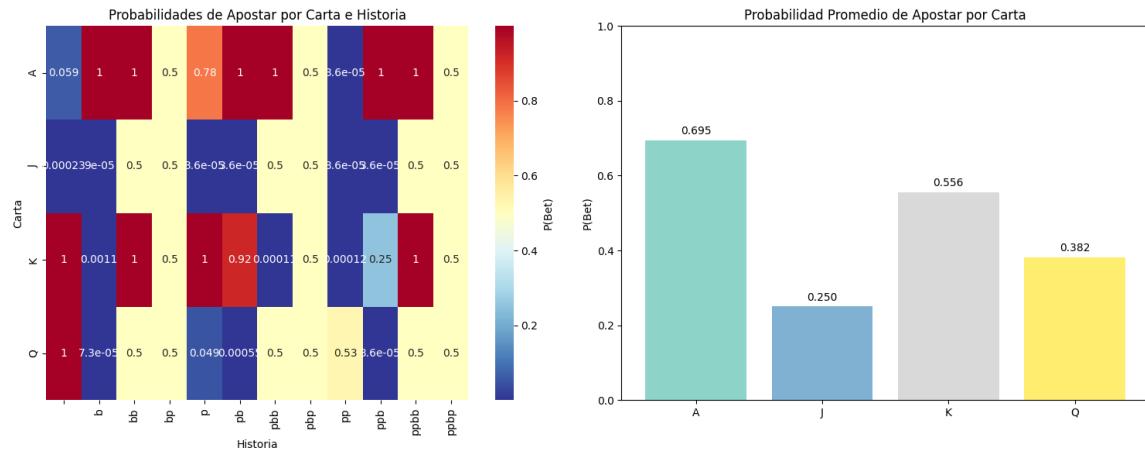
#### 4.2.2.6 Entrenamiento largo (100.000 iteraciones) y Corrida de estudio de 10.000 partidas

##### Evolucion de la politica:



- Carta J: Las probabilidades de apostar se mantienen prácticamente en cero en todos los nodos. Esto es coherente con una política que evita arriesgar con la carta más baja.
- Carta Q: Se observa una ligera evolución en algunos nodos (e.g., 1pp), pero la mayoría de las probabilidades se estabilizan cerca de 0.5, sugiriendo una estrategia más mixta con esta carta.
- Carta K: La política de apostar es clara en varios nodos (e.g., 2pb, 2pp), mostrando valores cercanos a 1, lo que indica que el agente apuesta agresivamente con esta carta.
- Carta A: Las probabilidades se estabilizan en valores altos ( $>0.75$ ) en muchos nodos, reflejando un comportamiento fuertemente agresivo con la mejor carta del juego, lo cual es esperable.

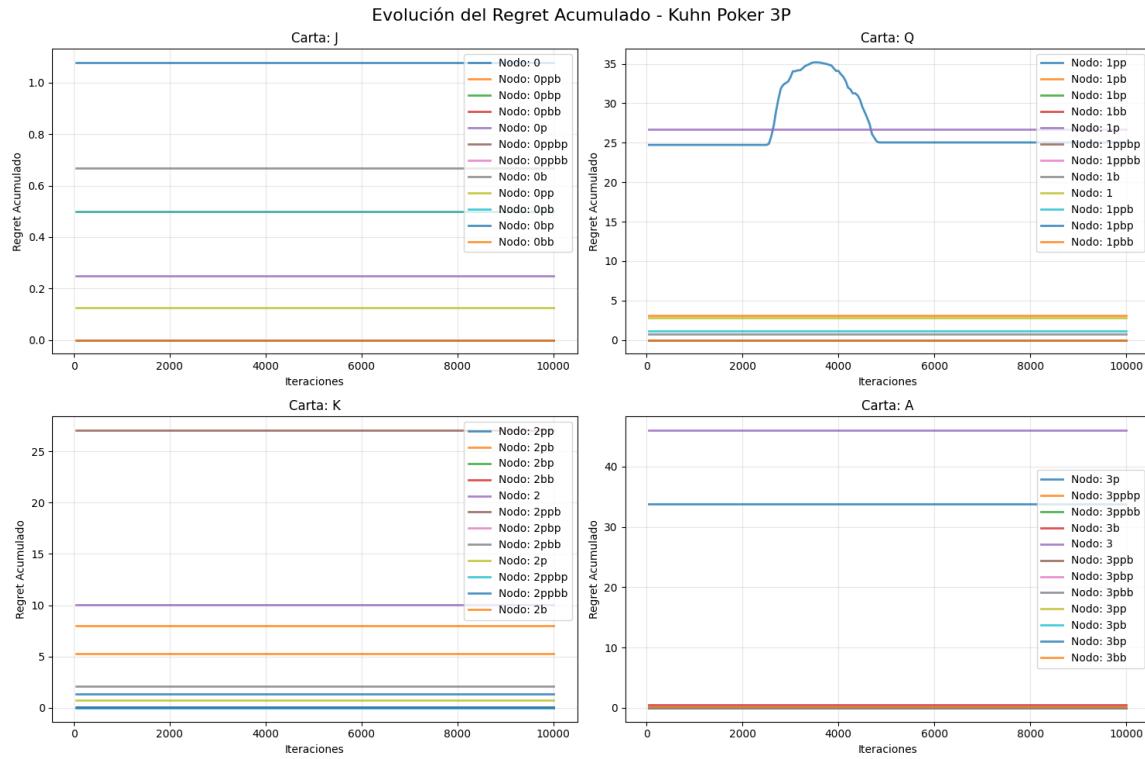
## Heatmap de decisiones y Promedio de probabilidad de apostar por carta:



Se observa una política coherente con el valor de las cartas. La A tiene valores cercanos a 1 en casi todos los contextos. K también mantiene valores altos, aunque algo más variados. Q presenta valores más mixtos, mientras que J tiene probabilidades muy bajas en casi todos los nodos, reflejando prudencia.

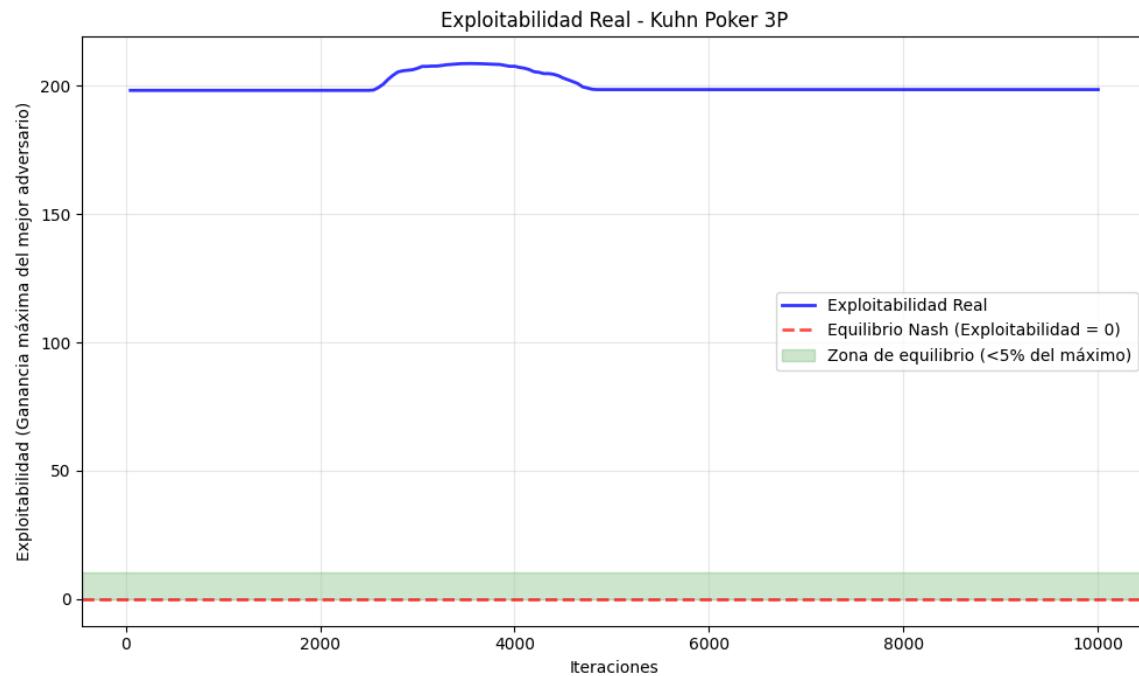
El orden es coherente con el valor de las cartas, y las diferencias están bien marcadas.

## Evolución del regret acumulado:



- Carta J: Los regrets son bajos y estables, lo que indica que las decisiones con J no generan mucha pérdida de oportunidad.
- Carta Q: Se evidencia un pico temporal en 1pp, lo que sugiere que el agente estuvo probando estrategias subóptimas en ese nodo antes de corregirse.
- Carta K y A: Los regrets más altos se concentran en nodos donde se toman decisiones críticas. Esto sugiere que el agente aún enfrenta dilemas estratégicos puntuales, pero en general los valores son estables tras 100k iteraciones.

## Explotabilidad real



La explotabilidad aumenta con el entrenamiento. Tras una pequeña fluctuación en la mitad del entrenamiento, el valor final de explotabilidad es más alto que en fases anteriores, y se aleja de la zona de equilibrio.

Esto indica que el agente no está convergiendo hacia un equilibrio de Nash, y que las estrategias aprendidas resultan altamente explotables por un adversario óptimo.

#### 4.2.2.7 Resumen

A lo largo de los tres bloques se evaluó el desempeño de un agente CFR en Kuhn Poker para 3 jugadores, con diferentes niveles de entrenamiento (1.000, 10.000 y 100.000 iteraciones) y torneos de validación de distinta longitud. Los resultados muestran un patrón claro: el aumento en la cantidad de iteraciones no conduce necesariamente a una mejora en la calidad de las estrategias aprendidas.

- Estabilidad de las políticas: Las políticas se estabilizan rápidamente, incluso en el primer bloque. Sin embargo, esta estabilidad no implica convergencia hacia una solución óptima, ya que se observa comportamiento rígido en muchas decisiones (por ejemplo, siempre pasar con J), lo cual se mantiene incluso con mayor entrenamiento.
- Regret acumulado: El regret deja de decrecer desde etapas tempranas. En los tres bloques, muchos nodos presentan valores planos, lo cual sugiere estancamiento en la actualización de estrategias. Solo algunos nodos asociados a la carta Q muestran oscilaciones puntuales.
- Probabilidades de apuesta por carta: Se mantiene constante el patrón de mayor agresividad con A y K, comportamiento mixto con Q y pasividad con J. El promedio de apuesta por carta prácticamente no cambia entre bloques, lo cual reafirma que el agente no continúa aprendiendo tras cierto punto.
- Exploitabilidad: Contrariamente a lo esperado, la exploitabilidad no disminuye con el entrenamiento. Por el contrario, aumenta en los bloques 2 y 3, alejándose cada vez más de la zona de equilibrio. Esto indica que las políticas aprendidas resultan más vulnerables frente a oponentes óptimos, probablemente debido a una sobreajuste a estrategias subóptimas propias.

En conjunto, estos resultados reflejan que el proceso de entrenamiento no está conduciendo a una mejora progresiva del agente. Esto podría deberse a un número insuficiente de iteraciones para un entorno tan complejo como el de 3 jugadores, a una mala configuración de los parámetros del CFR, o a una falta de mecanismos que promuevan exploración efectiva en fases avanzadas del entrenamiento.

## 5 Conclusiones generales

### 5.1 Juegos alternados de información perfecta

- Complejidad del entorno

TicTacToe es un juego completamente resuelto, de reglas simples y espacio de estados reducido, lo que permite a algoritmos clásicos como MinMax obtener un rendimiento casi óptimo con configuraciones de baja profundidad. Nocca Nocca, en cambio, presenta una estructura mucho más rica y compleja: mayor espacio de estados, reglas de movimiento dinámicas y un componente posicional más fuerte. Esto incrementa considerablemente el costo computacional y requiere una mayor sofisticación en el diseño de los agentes.

- Eficacia de los algoritmos

En TicTacToe, tanto MinMax como MCTS logran rendimientos cercanos al óptimo. La diferencia entre ellos es marginal y, en muchos casos, las partidas entre agentes bien configurados terminan en empate.

En Nocca Nocca, MCTS se destaca por su capacidad de adaptación y su habilidad para manejar entornos no triviales. Su naturaleza estocástica le permite detectar patrones estratégicos más allá de lo que puede alcanzar MinMax a profundidades razonables. MinMax es competitivo en configuraciones controladas, pero su rendimiento se estanca al no poder explorar con suficiente profundidad.

- Rol de la función de evaluación

En TicTacToe, la función de evaluación fue dada por la catedra.

En Nocca Nocca, se observó como la función de evaluación afecta directamente la calidad de las decisiones y permite que incluso búsquedas superficiales (como en MinMax de baja profundidad) generen buenos resultados si la evaluación está bien calibrada.

- Trade-offs computacionales

En TicTacToe, los algoritmos pueden configurarse para responder en tiempo real sin gran pérdida de desempeño. Esto no es así en Nocca Nocca, donde incluso

pequeños incrementos de parámetros (profundidad o número de simulaciones) pueden multiplicar el tiempo de ejecución. Esto obliga a tomar decisiones más cuidadosas según los recursos disponibles.

- Robustez y generalización

MCTS se muestra como un enfoque más robusto y generalizable a distintos entornos. Supera de forma consistente a agentes aleatorios en ambos juegos, incluso con configuraciones no optimizadas.

MinMax depende más del entorno: brilla en escenarios acotados como TicTacToe, pero requiere ayudas externas (funciones de evaluación eficientes) para ser competitivo en Nocca Nocca.

- Diseño del agente óptimo

En TicTacToe, el agente óptimo se obtiene afinando profundidad o cantidad de simulaciones, con cambios marginales entre distintas configuraciones una vez que se alcanza cierto umbral.

En Nocca Nocca, no existe una única solución óptima: el mejor agente depende del entorno, de los recursos disponibles y de cómo se combinan las piezas del diseño (algoritmo + evaluación + parámetros).

En Síntesis:

TicTacToe permite evaluar y comparar algoritmos en un entorno controlado y determinístico, ideal para validar conceptos básicos.

Nocca Nocca, en cambio, sirve como banco de pruebas para analizar el desempeño de agentes en situaciones más realistas, donde las decisiones estratégicas, la eficiencia computacional y la adaptabilidad son determinantes. El salto de complejidad entre ambos juegos resalta la importancia de diseñar agentes flexibles, especialmente cuando se enfrenta un entorno con reglas ricas y espacio de estados más amplio.

## 5.2 Juegos alternados de información perfecta

El entrenamiento del agente CFR sobre Kuhn Poker de 2 y 3 jugadores permite observar diferencias significativas en el comportamiento del algoritmo y en la dinámica del aprendizaje, atribuibles tanto a la estructura del juego como al espacio de estrategias involucrado.

- Convergencia y estabilidad del aprendizaje:

En el caso de 2 jugadores, el agente CFR converge rápidamente hacia estrategias próximas al equilibrio de Nash. Se observa una clara disminución progresiva del regret y una exploitabilidad decreciente, lo cual indica un aprendizaje efectivo.

En contraste, en 3 jugadores, aunque las políticas se estabilizan rápidamente, esta estabilidad no representa mejora. El regret se mantiene constante en muchos nodos y la exploitabilidad no solo no disminuye, sino que tiende a aumentar, lo cual revela que el aprendizaje se estanca lejos del equilibrio.

- Complejidad del espacio estratégico:

Kuhn Poker 3P es sustancialmente más complejo: el número de nodos de decisión, combinaciones de historia y posibles interacciones estratégicas crece exponencialmente. Esto genera un entorno más difícil para CFR, que depende del muestreo constante de contraestrategias.

Este aumento de complejidad parece llevar al agente a sobreajustarse a patrones subóptimos, limitando la capacidad de refinar sus políticas a largo plazo.

- Patrón en las estrategias aprendidas:

En ambos casos, las políticas resultantes exhiben comportamientos coherentes con la fuerza de la carta (más apuestas con cartas altas, más pasividad con cartas débiles), pero en 3 jugadores estas tendencias son más extremas y rígidas, y no necesariamente reflejan decisiones óptimas dadas las múltiples dependencias estratégicas.

- Impacto del número de iteraciones y tamaño del torneo:

En 2 jugadores, aumentos en iteraciones y partidas refuerzan la calidad de la política.

En 3 jugadores, más entrenamiento no mejora el desempeño y puede incluso empeorarlo en términos de vulnerabilidad estratégica (mayor exploitabilidad), señal de que el agente no logra generalizar ni corregir desviaciones.

#### Conclusión final

El algoritmo CFR resulta efectivo y confiable en entornos de dos jugadores como Kuhn Poker 2P, donde logra estrategias cercanas al equilibrio. Sin embargo, su desempeño se degrada significativamente en juegos de tres jugadores, donde la complejidad adicional genera estancamiento, sobreajuste y estrategias explotables. Esto resalta la necesidad de incorporar técnicas más robustas (como CFR+ o Deep CFR) o mecanismos que favorezcan exploración y diversidad estratégica en entornos multijugador.

## **6 Anexo**

Puede encontrarse las notebooks utilizada a traves del siguiente enlace de GitHub

[https://github.com/valeriaeskenazi/Obligatorio\\_2\\_sistemas\\_multiagentes](https://github.com/valeriaeskenazi/Obligatorio_2_sistemas_multiagentes)

[https://github.com/valeriaeskenazi/Obligatorio\\_2\\_sistemas\\_multiagentes.git](https://github.com/valeriaeskenazi/Obligatorio_2_sistemas_multiagentes.git)

