



**UNIVERSIDAD DEL VALLE
SEDE TULUÁ**

INTRODUCCIÓN A LA INTELIGENCIA ARTIFICIAL

**PROYECTO FINAL
AJEDREZ LOCO**

**JUAN SEBASTIAN GONZALEZ CAMACHO
ANDRES FELIPE RUIZ BURITICA
KEVIN STEVEN VICTORIA OSPINA**

**TULUÁ
8 DE JULIO DEL 2023**

ÍNDICE

ÍNDICE.....	1
INTRODUCCIÓN.....	2
IMPLEMENTACIÓN.....	3
INTERFAZ GRÁFICA.....	3
REGLAS Y MOVIMIENTOS DEL AJEDREZ.....	5
INTELIGENCIA ARTIFICIAL.....	6
CONCLUSIONES.....	9

INTRODUCCIÓN

En este proyecto final de Introducción a la Inteligencia Artificial, se explorará la creación de una variedad del juego de ajedrez: el ajedrez loco, aplicando conceptos como árboles, algoritmos minimax, poda alfa y beta. Se utilizarán estos conocimientos para desarrollar estrategias y tácticas de inteligencia artificial, desafiando nuestras habilidades de programación y ampliando nuestra comprensión de la IA en el ámbito de los juegos.

IMPLEMENTACIÓN

INTERFAZ GRÁFICA

Para la GUI utilizamos la biblioteca **Pygame**¹ para la interfaz gráfica, la cual es una biblioteca de programación de juegos que proporciona funcionalidades para la creación de gráficos, animaciones y sonido en juegos.

El juego se basa en una representación de nodos que forman el tablero de ajedrez. Cada nodo representa una celda en el tablero y contiene información sobre su posición, color y contenido. La función **make_grid** crea una cuadrícula de nodos que forman el tablero de ajedrez. La función **draw_grid** dibuja las líneas de la cuadrícula en la ventana del juego.

La función **update_display** actualiza la ventana del juego para mostrar el tablero de ajedrez y las piezas en su posición actual. La función utiliza una matriz para almacenar la información sobre las piezas en el tablero y luego dibuja las piezas en la ventana del juego.

La función **find_node** encuentra la posición del nodo en el tablero de ajedrez a partir de una posición dada en píxeles en la ventana del juego.

La función **main** es la función principal que ejecuta el juego de ajedrez. Utiliza un bucle principal para manejar los eventos del juego, como los clics del mouse. Cuando el jugador realiza un movimiento, se actualiza el tablero de ajedrez y luego es el turno de la IA. La función **ia_turn** realiza el movimiento de la IA utilizando la función **machine_move** que determina el movimiento de la IA.

Las imágenes de las fichas y los colores de las casillas los tomamos prestados de una de las plataformas de juego de ajedrez más conocidas, **Chess**². Nos pareció buena idea usarlos ya que son elementos gráficos bastante elegantes con un toque moderno.

¹ <https://www.pygame.org/news>

² <https://www.chess.com/es>



Figura 1: Muestra de la interfaz gráfica.

Para lograr capturar los eventos del ratón, se implementó un manejo de eventos dentro de un bucle principal. Este bucle se encarga de detectar y manejar los diferentes eventos del mouse, como clics y movimientos.

Cuando se detecta un evento de clic del mouse, se realiza lo siguiente:

1. Se obtiene la posición del mouse en píxeles dentro de la ventana del juego.
2. Utilizando la función **find_node**, se encuentra la posición correspondiente del nodo en el tablero de ajedrez. Esta función mapea la posición en píxeles a una posición en la notación del ajedrez, como "a1", "e5", etc.
3. Se verifica si hay una ficha capturada. En caso de haber una ficha capturada, se intenta colocarla en la posición seleccionada. Se valida que la posición esté vacía y se realiza la colocación. En caso de que la posición ya esté ocupada, se muestra un mensaje de error indicando que la posición no es válida.
4. Si no hay una ficha capturada, se verifica si el movimiento actual es el inicio de un nuevo movimiento o la continuación de uno existente. Si es el inicio de un nuevo movimiento, se verifica que la ficha seleccionada sea válida (es decir, pertenezca al jugador actual). Si no es válida, se muestra un mensaje de error.
5. Si es la continuación de un movimiento existente, se completa el movimiento y se verifica su validez. Si el movimiento es válido, se realiza en el tablero y se actualiza la visualización. En caso contrario, se muestra un mensaje de error indicando que el movimiento no es válido.

Al utilizar esta lógica de eventos del mouse, se logra que los jugadores puedan seleccionar y mover las fichas del ajedrez de manera interactiva en la ventana del juego. Los eventos del mouse permiten una experiencia de juego más dinámica y fluida, ya que los jugadores pueden interactuar directamente con el tablero y realizar movimientos con facilidad.

REGLAS Y MOVIMIENTOS DEL AJEDREZ

La biblioteca **chess**³ proporciona funcionalidades esenciales para la implementación de las reglas y movimientos permitidos del ajedrez. Al iniciar el juego, se crea un objeto **board** que representa el estado actual del tablero y contiene información sobre la posición de las piezas.

Utilizando las funciones y métodos provistos por la biblioteca **chess**, se pueden realizar diversas operaciones en el tablero. Por ejemplo, la función **legal_moves** genera todos los movimientos legales en el tablero actual, permitiendo verificar la validez de los movimientos realizados por los jugadores. Además, la función **piece_at** permite obtener información sobre las piezas en una posición específica del tablero.

La lógica del juego se implementa mediante la combinación de estas funciones y métodos con la lógica específica del proyecto. Por ejemplo, al recibir un movimiento del jugador, se verifica su validez utilizando **legal_moves**, y si es válido, se aplica utilizando **push**. Además, se pueden utilizar otras funciones como **is_checkmate** para verificar el estado de jaque mate, **is_stalemate** para verificar un empate por ahogamiento, y **is_check** para verificar si el rey está en jaque.

Esta librería utiliza letras para identificar cada ficha, las mayúsculas son las fichas blancas y las minúsculas las fichas negras.

R: Torre blanca.
N: Caballo blanco.
B: Alfil blanco.
Q: Reina blanca.
K: Rey blanco.
P: Peón blanco.

r: Torre negra.
n: Caballo negro.
b: Alfil negro.
q: Reina negra.
k: Rey negro.
p: Peón negro.

³ <https://python-chess.readthedocs.io/en/latest/>

r	n	b	q	k	b	n	r
p	p	p	p	p	p	p	p
.
.
.
.
P	P	P	P	P	P	P	P
R	N	B	Q	K	B	N	R

Figura 2: Demostración de la representación del tablero.

INTELIGENCIA ARTIFICIAL

El objetivo de la inteligencia artificial en este juego de ajedrez es permitir que la máquina tome decisiones estratégicas y realice movimientos óptimos en función del estado actual del tablero. En este caso, se ha implementado un algoritmo de búsqueda con poda alfa-beta para determinar los movimientos de la IA.

El algoritmo utiliza una evaluación heurística⁴ para asignar valores a las diferentes configuraciones del tablero. Estos valores se dividen en dos categorías: valores materiales y valores posicionales. Los valores materiales representan la importancia relativa de cada pieza en función de su tipo (peón, caballo, alfil, torre, dama y rey). Los valores posicionales, por otro lado, evalúan la calidad de la ubicación de las piezas en el tablero.

Para cada movimiento posible en un determinado estado del tablero, se realiza una búsqueda en el árbol de juego utilizando el algoritmo de poda alfa-beta. Este algoritmo evalúa de manera recursiva diferentes configuraciones del tablero hasta alcanzar una cierta profundidad de búsqueda.

En cada nivel de la búsqueda, se alternan los roles de **maximizador** y **minimizador**. El maximizador busca maximizar el valor de evaluación del tablero, mientras que el minimizador busca minimizarlo. La poda alfa-beta permite eliminar ramas de búsqueda que no afectarán la decisión final, lo que resulta en una mejora significativa en la eficiencia del algoritmo.

⁴ <https://www.freecodecamp.org/news/simple-chess-ai-step-by-step-1d55a9266977/>

Durante la búsqueda, se evalúa cada configuración del tablero utilizando la función **evaluate_board** o **evaluate_board_alt**, dependiendo del contexto. Estas funciones asignan un valor numérico al estado del tablero en función de los valores materiales y posicionales de las piezas presentes.

Una vez que se completa la búsqueda en el árbol de juego hasta la profundidad especificada, se selecciona el movimiento con el valor de evaluación más alto para ser realizado por la IA. Este movimiento se devuelve como resultado de la función **machine_move**.

Además, se ha implementado la funcionalidad de colocar las fichas robadas por parte de la máquina. Para esto, se utiliza la función **put_piece**, que realiza una búsqueda similar en el árbol de juego para seleccionar la mejor casilla vacía en la que colocar la ficha robada.




	10		-10
	30		-30
	30		-30
	50		-50
	90		-90
	900		-900

Figura 3: Ilustración del valor material de las fichas.

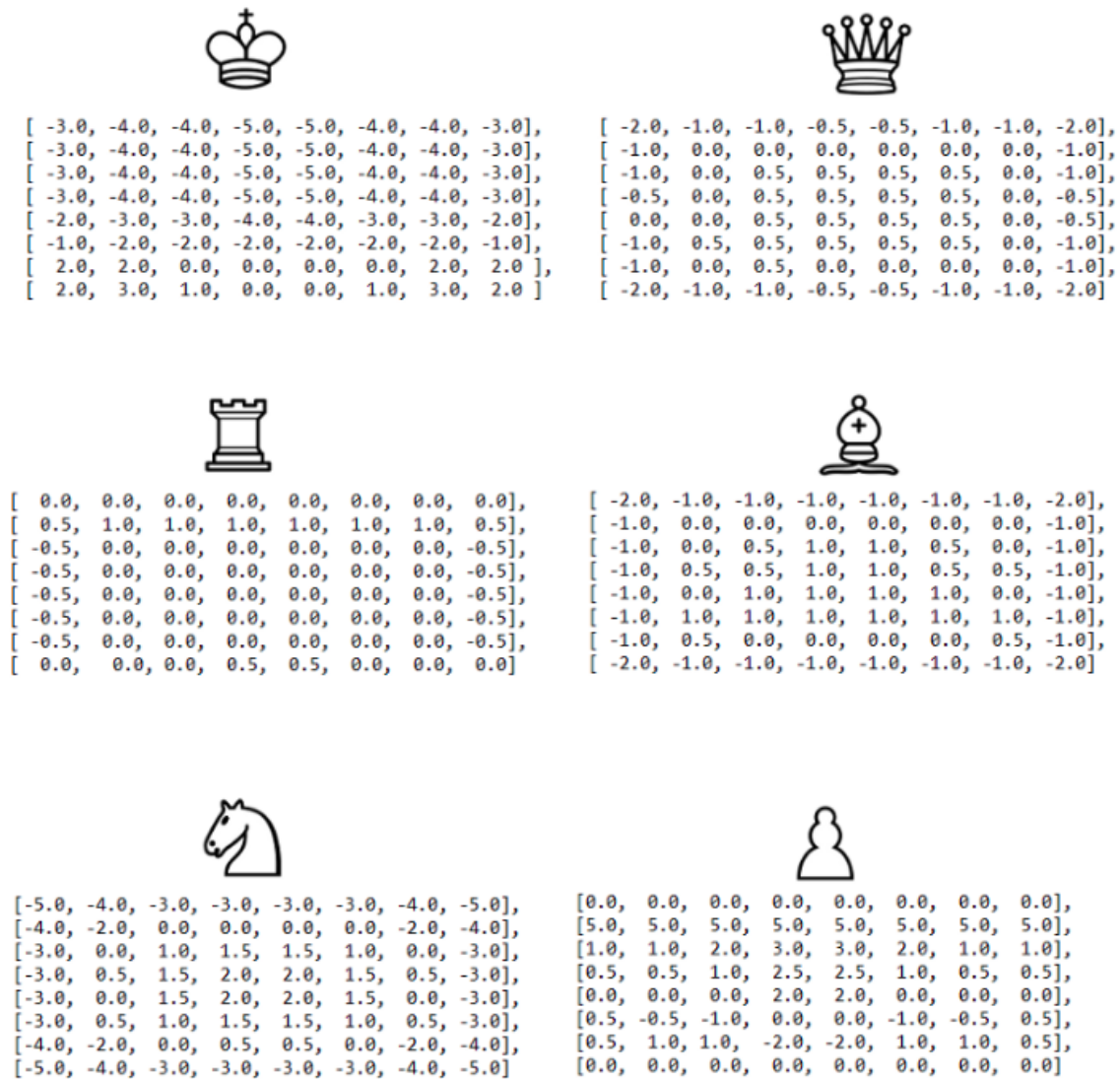


Figura 4: Ilustración del valor posicional de las fichas.

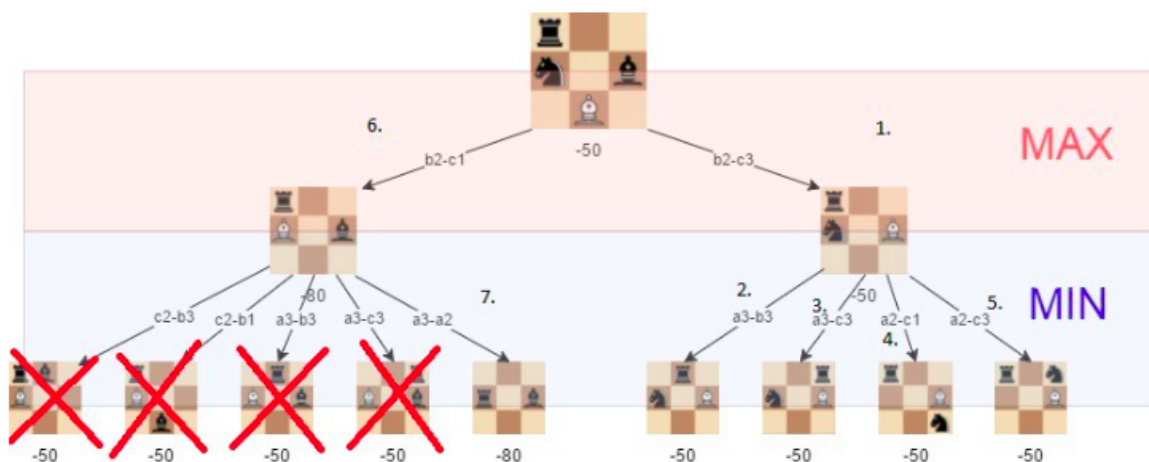


Figura 5: Ilustración de la poda alpha-beta en el minmax.

CONCLUSIONES

La función de evaluación heurística desempeñó un papel fundamental en la toma de decisiones de la IA. Se utilizaron valores materiales y posicionales para asignar puntuaciones a las diferentes configuraciones del tablero. Esto permitió evaluar de manera aproximada la calidad de una posición y guiar las decisiones de la IA hacia movimientos más favorables.

Se logró una integración exitosa entre el algoritmo de búsqueda y la función de evaluación, lo que permitió que la IA tomara decisiones estratégicas y realizara movimientos que maximizaran sus posibilidades de éxito en el juego.

Se implementó un algoritmo de búsqueda con poda alfa-beta, que permitió realizar una exploración eficiente del árbol de juego hasta una cierta profundidad. La poda alfa-beta resultó ser una técnica crucial para reducir el número de nodos explorados y mejorar el rendimiento del algoritmo. Sin embargo, cabe mencionar que el proyecto tiene algunas limitaciones. La profundidad de búsqueda elegida puede afectar el rendimiento y la precisión de la IA. Una profundidad mayor permitiría una evaluación más precisa, pero también requeriría más tiempo de cómputo. Además, la función de evaluación heurística utilizada puede ser mejorada y refinada para considerar más aspectos del juego.

De manera general, el proyecto ha demostrado el potencial de las técnicas de inteligencia artificial en el juego de ajedrez. A través de la implementación de un algoritmo de búsqueda con poda alfa-beta y una función de evaluación heurística, se logró crear una IA capaz de competir contra jugadores humanos y tomar decisiones estratégicas de forma autónoma.