Proyecto 2

Ajedrez de Alicia

Inteligencia Artificial

Elaborado por:

Rodas Arango, JUAN MANUEL - 2259571

García Castañeda, ALEX – 2259517

Gómez Agudelo, JUAN SEBASTIÁN – 2259474

Henao Aricapa, STIVEN – 2259603

Docente:

Triana Madrid, JOSHUA

Sede Tuluá

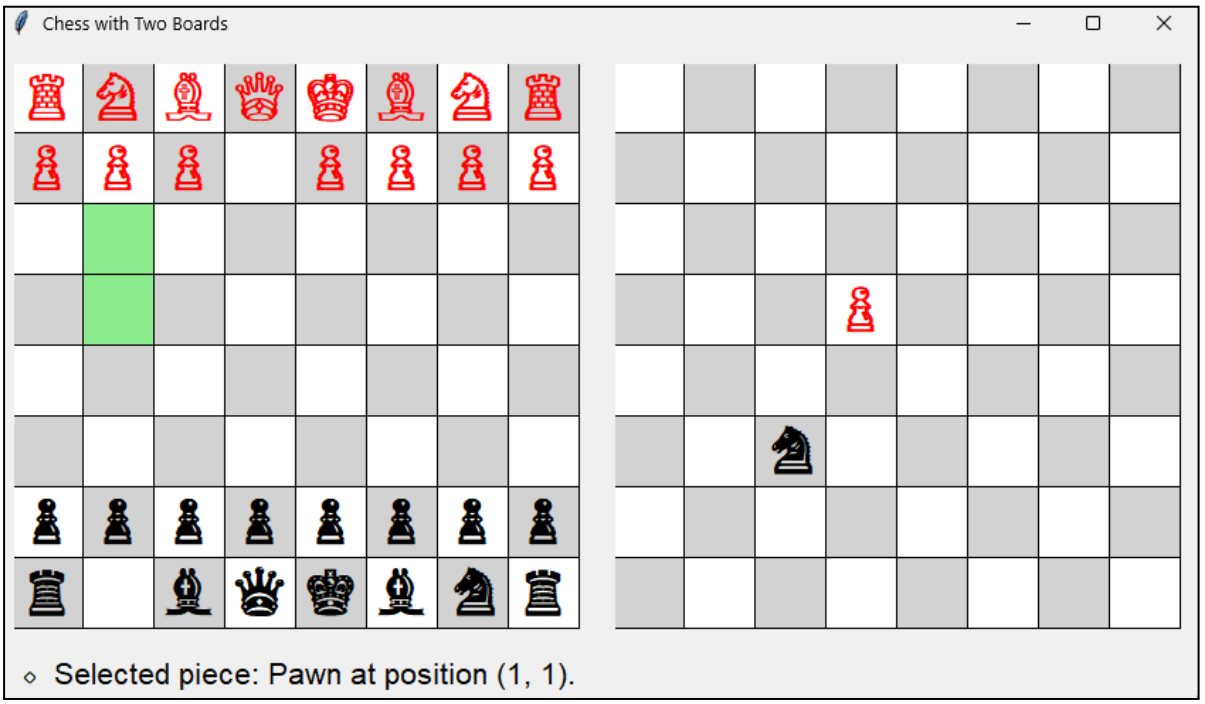
Diciembre 2024

Índice

1. Introducción **3**
2. Estructura del Proyecto **4**
3. Implementación Destacada: El Algoritmo Minimax **5**
3.1 Descripción del Minimax **5**
3.2 Funcionamiento del Minimax **5**
4. Características Adicionales **6**
5. Pruebas y Resultados **6**

1. Introducción

El presente proyecto tiene como objetivo desarrollar una aplicación de ajedrez funcional, con soporte para jugar uno a uno o contra una inteligencia artificial (IA). La lógica central del juego se basa en el algoritmo **Minimax**, que permite simular movimientos y tomar decisiones óptimas para un jugador virtual. Este informe detalla los módulos principales del proyecto (app.py, board.py, GUI.py, pieces.py), destacando sus características, la implementación del Minimax, y otros elementos relevantes.



2. **Estructura del Proyecto**

1. **app.py: Controlador Principal**
   Este módulo actúa como el punto de entrada del programa, gestionando la ejecución del juego y la interacción entre el usuario, la interfaz gráfica (GUI), y la lógica del tablero.
   - Gestiona los turnos del jugador y de la IA (si está habilitada).
   - Permite inicializar el tablero y las piezas en sus posiciones estándar.
   - Incluye funciones específicas para habilitar/deshabilitar el algoritmo Minimax según las preferencias del usuario.
2. **board.py: Lógica del Tablero**
   Este módulo define la estructura del tablero de ajedrez y las interacciones entre las piezas.
   - Implementa métodos para agregar, mover y eliminar piezas.
   - Soporta un modelo de tablero ampliado, considerando posibles extensiones para variantes del juego.
   - Incluye validaciones para movimientos legales según las reglas del ajedrez.
3. **pieces.py: Representación de las Piezas**
   Este archivo define las clases individuales para cada tipo de pieza: Rey, Reina, Torre, Alfil, Caballo, y Peón.
   - Cada clase tiene reglas específicas para sus movimientos y capacidades (por ejemplo, el enroque para el Rey o la promoción de peones).
   - Incluye una representación jerárquica que permite manejar a las piezas de forma uniforme en el tablero.
4. **GUI.py: Interfaz Gráfica de Usuario**
   Este módulo proporciona una interfaz visual para interactuar con el juego.
   - Representa gráficamente el tablero y las piezas en sus posiciones actuales.
   - Resalta los movimientos legales disponibles para cada pieza seleccionada.
   - Permite alternar entre los movimientos del usuario y los generados por la IA.

3. Implementación Destacada: El Algoritmo Minimax

3.1 Descripción del Minimax

El algoritmo **Minimax** es el núcleo de la lógica de IA del proyecto. Su implementación permite que la máquina anticipe movimientos futuros, evalúe posibles escenarios, y tome decisiones estratégicas.

- **Objetivo:** Maximizar la ganancia del jugador controlado por la IA, minimizando al mismo tiempo las oportunidades del oponente.
- **Heurísticas Utilizadas:**
    1. **Valor Material:** Cada pieza tiene un valor asociado según su relevancia en el juego:
        - Peón: 1 punto.
        - Caballo: 3 puntos.
        - Alfil: 3 puntos.
        - Torre: 5 puntos.
        - Reina: 9 puntos.
        - Rey: 100 puntos.
    2. La IA prioriza capturas de piezas de mayor valor e identifica oportunidades para mejorar su posición material.
    3. **Posicionamiento Estratégico:**
       Las casillas del tablero tienen valores asociados para cada tipo de pieza.
        - Los peones obtienen más valor al acercarse al extremo del tablero, ya que pueden ser promovidos.
        - El centro del tablero es crucial para piezas como el Caballo y el Alfil, ya que aumenta sus posibilidades de movimiento.
        - Estas estrategias están inspiradas en las prácticas recomendadas por jugadores profesionales.
- **Poda Alpha-Beta:**
  Para optimizar el rendimiento, la implementación incluye poda Alpha-Beta, que reduce el número de movimientos evaluados descartando ramas irrelevantes.

3.2 Funcionamiento del Minimax

1. El algoritmo evalúa todas las posibles jugadas legales del jugador actual.
2. Calcula el puntaje resultante basado en las heurísticas mencionadas.
3. Simula los movimientos del oponente, asumiendo que también tomará decisiones óptimas.
4. Selecciona el movimiento que maximiza el puntaje para la IA mientras minimiza las opciones del oponente.

4. Características Adicionales

1. **Reglas de Ajedrez Implementadas:**
    - Movimientos estándar de todas las piezas.
    - Promoción automática de peones al llegar al extremo opuesto del tablero.
    - Enroque del Rey y Torre, considerando condiciones legales.
    - Detección de jaque y jaque mate.
2. **Interfaz Gráfica:**
    - El tablero es interactivo y permite arrastrar piezas para realizar movimientos.
    - Resalta visualmente las casillas disponibles para cada pieza seleccionada.
3. **Turnos y Modos de Juego:**
    - Modo uno a uno: Ambos jugadores controlan manualmente sus movimientos.
    - Modo contra IA: Un jugador humano se enfrenta al algoritmo Minimax.

5. Pruebas y Resultados

El proyecto fue probado exhaustivamente en diferentes escenarios, destacando lo siguiente:

- La IA es capaz de anticipar movimientos y castigar errores estratégicos del oponente.
- Se observaron mejoras significativas en el rendimiento al implementar poda Alpha-Beta, especialmente en tableros con configuraciones complejas.
- Las reglas de ajedrez se cumplen correctamente, incluyendo condiciones especiales como enroques y jaques.