Proyecto 2: Othello

Palacios Gómez Ernesto Rubén, 312280622
Flores González Luis Brandon, 312218342
Peto Gutierrez Emmanuel, 414008117
Luis Enrique Cortez Flores, 311209219

Introducción

Siempre ha sido un reto para el hombre lograr que las máquinas piensen igual que nosotros, o al menos simular tal situación. Desde sus orígenes la inteligencia artificial se relacionó con juegos como el ajedrez y las damas, probablemente debido a que los juegos de mesa representan modelos de situaciones reales en las que hay que calcular, solucionar problemas, tomar decisiones, corregir errores, recordar, etc. La inteligencia artificial en juegos se concentra en el estudio estratégico de la toma de decisiones, aplicada mediante un agente capaz de evaluar y actuar con ciertos principios de optimización y coherencia para cumplir una meta o un propósito. Las técnicas de búsqueda son métodos fundamentales para resolver problemas y en particular las heurísticas se aplican a problemas específicos y con el fin de hallar soluciones. Las heurísticas exploran una pequeña porción del espacio de solución, donde se cree que buenas soluciones se pueden hallar.

En este proyecto presentamos el caso particular de la aplicación de la inteligencia artificial en el juego de Othello.

Definición del problema

Othello o Reversi es un juego de estrategia de dos jugadores. Se juega sobre un tablero monocolor de 64 casillas, de 8 por 8. Los jugadores disponen de 64 fichas bicolores.

El objetivo del juego es tener más fichas de nuestro color que el adversario al final de la partida pero puede ocurrir el caso de llegar a un empate si ambos jugadores se quedan sin futuras jugadas y si el número de fichas propietarias es igual al del contrincante. El final se produce cuando ningún jugador puede realizar más movimientos.

El estado inicial del juego es donde se colocan dos fichas negras y dos fichas blancas(dependiendo el color) al centro del tablero.

En su turno de juego, el jugador debe colocar una ficha de su color sobre una casilla vacía del tablero, adyacente a una ficha contraria. Al colocar su ficha, se debe rodear(flanquear) una o varias fichas contrarias entre la ficha que se coloca y una ficha del mismo color, ya colocada sobre el tablero. Este flanqueo puede ocurrir en las ocho direcciones si alguna de estas cumple con la condición ya dada. A continuación se voltean de su color las fichas que acaba de flanquear. Las fichas no se retiran del tablero ni se mueven de una casilla a otra.

El problema a resolver con nuestro programa es hacer que la máquina juegue othello de manera óptima pero sin agotar los recursos computacionales.

Descripción de la propuesta e implementación

Minimax.

El algoritmo minimax se utiliza para obtener la decisión más óptima en un juego para el jugador que llamaremos MAX, al otro jugador le llamaremos MIN. Este consiste en expandir un árbol donde se obtienen todas las jugadas posibles, donde cada nodo es un estado del juego y sus hijos son todas las jugadas que se pueden obtener en un turno, salvo que haya terminado el juego. Cuando se llega a las hojas del árbol, o sea al final del juego, se evalúa el juego con una función de utilidad y se le asigna un número. Luego se asigna un valor a cada nodo no terminal con la siguiente regla:

- Si tira MAX, entonces elegirá a su hijo con un valor mayor.
- Si tira MIN, entonces elegirá a su hijo con un valor menor.

Cuando el nodo raíz, que es de MAX, tenga su número, tomará la acción que lo lleva al nodo hijo con un valor más grande.
Con este algoritmo se asume que ambos jugadores juegan de forma óptima. El problema con este algoritmo es que crece de forma exponencial, y en juegos con muchas posibles jugadas y que terminan en muchos turnos se vuelve un problema intratable. Lo que se hace en juegos de este tipo minimizar el tiempo de ejecución y la memoria utilizada. Esto se hace haciendo una optimización del algoritmo llamado “poda alfa-beta” donde no se tienen que revisar todas las ramas si ya se ha encontrado un resultado más óptimo, y otra optimización es no llegar al final del juego, sino evaluar un nodo con una función que le asigna un número al estado del juego, donde un número mayor significa que MAX tiene mayor probabilidad de ganar.

Agente

El tipo de agente implementado para lograr responder de la mejor manera al tiro del contrincante, es decir, lograr la mejor jugada posible dado un razonamiento es un agente basado en objetivos llamado **agente resolvente-problemas** ya que este conduce al estado deseable que es el tablero con una tirada que aumenta la posibilidad de éxito contra la jugada previa del contrincante.

Además este hace una búsqueda informada ya que tenemos la medida de utilidad donde esta consiste en la paridad, movilidad, estabilidad y esquinas capturas en un estado.

Definiremos el REAS con una tabla:

Caso	Rendimiento	Entorno	Actuadores

Agente que juega Othello.	Ganar el juego.	Tablero de 8x8 con fichas que le pertenecen al agente y otras fichas que le pertenecen al rival.	Tirar fichas.

Implementación.

Para implementar este programa se utilizó el algoritmo minimax sin poda, con un árbol incompleto (profundidad 4 por cada tiro), por ello creamos una función de evaluación que es una suma de 4 heurísticas. La función de evaluación devuelve un valor más grande si el estado favorece a la máquina y menor si favorece al jugador humano. Las heurísticas usadas fueron las siguientes:

- **Paridad:** es un conteo de la cantidad de fichas que hay hasta el momento por jugador.
- **Movilidad:** cuenta las posibles jugadas siguientes que tiene cada jugador.
- **Esquinas:** cuenta las esquinas capturadas y les da un alto valor, pues estas no se pueden cambiar de color una vez tomadas.
- **Estabilidad:** es una diferencia entre las fichas estables y las fichas inestables. Las fichas estables son las que ya no se pueden cambiar de color y estas son las que están en las esquinas, o hacen una línea horizontal o vertical del mismo color con la esquina, o bien son parte de un rectángulo del mismo color formado con la esquina. Las fichas inestables son las que se pueden cambiar de color en un turno.

Es importante mencionar que se crea un árbol nuevo y se aplica el algoritmo cada vez que la máquina va a tirar.

Complejidad.

Por cada turno se tienen entre 5 y 15 jugadas posibles, entonces el factor de ramificación medio es 10 aproximadamente. Cada vez que va a tirar el agente expandimos el árbol a una profundidad de 4, entonces b=10 y m=4. Debido a que en nuestra implementación primero creamos el árbol de posibles jugadas antes de ejecutar el algoritmo minimax, la complejidad espacial es el número de nodos en el árbol, lo cual es aproximadamente 10^5 -1, entonces la complejidad espacial es O(b^5).
Las heurísticas hacen comprobaciones sobre todo el tablero y este tiene 8^2 casillas, entonces haré la aproximación de que la complejidad de las heurísticas es de O(t^2). Las heurísticas solo se aplican a las hojas, que son aproximadamente 10^4 , entonces la complejidad en tiempo que utiliza minimax es O($b^4 x\ t^2$) donde t es la longitud del tablero, o sea aproximadamente $10^4 x\ 8^2$.

En este caso el agente juega a ganar, pues busca maximizar su número de fichas, tener más fichas estables(que no se puedan girar) y maximizar su cantidad de posibles movimientos por turno. El comportamiento no siempre es adecuado, pues el agente solo

puede ver 4 jugadas adelante y puede que en 4 movimientos busque maximizar su número de fichas, pero en este juego se pueden perder muchas fichas en pocos movimientos si el oponente juega mejor.

Conclusiones

En la actualidad se ha comprobado, mediante partidas con profesionales, que las máquinas han superado a los humanos en juegos de mesa (excepto en Go) por su capacidad de almacenamiento y de procesamiento. Pero aún con tanta memoria y procesador es difícil que una computadora juegue perfecto, así que aún se necesitan estrategias que utilizan los humanos para decidir cuándo es más probable ganar. En este caso comprobamos que es muy complejo evaluar muchas jugadas posteriores, por lo tanto no utilizamos una profundidad muy grande. Así que nuestro agente juega lo suficientemente bien para ganarle a un principiante pero no a un profesional, o en su defecto, a otra computadora que juegue en nivel profesional.

Referencias bibliográficas

RUSSELL, S. J.; NORVIG, P., (2004), *Inteligencia Artificial: Un enfoque moderno,* Segunda edición, Madrid, España, PEARSON EDUCACIÓN, S.A.

Sannidhanam, V.; Annamalai, M., An Analysis of Heuristics in Othello.

Lazard, E., Descubriendo el Othello.