29/Noviembre/2020

Matemáticas Aplicadas Y Ciencias de la Computación / MACC Entrega Final de Algoritmos y Estructura de datos 2020-II

Implementación del Algoritmo MiniMax en Triqui

Presentado por:

Natalia Katherine Rojas Suarez

Nicolas Dussan Castañeda

Miguel Angel Caicedo Carrasquilla

Lenguaje de Programación de Desarrollo del Proyecto: C++ 14.

Repositorio: https://github.com/miguelcaicedo4/Proyecto-Algoritmos-.git

Resumen ejecutivo:

Desde que somos pequeños jugamos diferentes cosas, los juegos varían según la época en la que te criaste, algunos de los más populares son: saltar la cuerda, escondidas, stop, golosa y demás. Algo que no a cambiado en las generaciones son nuestras ansias de ganar, cuando no lográbamos esto intentábamos empatar pero que jamás existiera la posibilidad de perder contra nuestro rival.

Para esto cada juego tenía sus pequeños trucos o estrategias para completar con satisfacción el objetivo, ganarle al otro. Para explicar esto mejor pensaremos en el ajedrez, cada pieza tiene sus respectivos movimientos, que en este caso serán nuestras reglas, con estas te defiendes, matas o avanzas. A este juego le puedes incrementar la dificultad agregándole reglas, como cierta cantidad de tiempo por jugada, o quitándole, como que, aunque llegue un peón al final del tablero no volverá la reina al mismo. Estas estrategias nos sirven para tener un juego mucho más organizado y equitativo para cada jugador. Existen jugadas que te garantizan un Jaque Mate, pero aun así estas dependen de muchas posibilidades y coincidencias que se tienen que dar para poder completarlas.

Nosotros quisimos explorar un poco más un juego que gran parte de la población mundial conoce con distintas variedades de nombres, el triqui. Este juego, como el ajedrez, tiene sus reglas a seguir, solo se puede ganar con una diagonal, columna o fila del mismo símbolo, solo pueden jugar dos personas, las cuales tienen X y O cada cual, para distinguir sus jugadas del contrincante, solo puede haber una figura por casilla. También debemos tener en cuenta el tablero, son solo 9 casillas posibles, eso nos da un máximo 9 turnos rotativos para jugar. Como podemos apreciar es muy simple la idea del juego, este posee, al igual que otros, jugadas que si las obtienes ganarías irrefutablemente.

Pero como podemos obtener estas jugadas si las posibilidades del triqui son muchísimas, ya que no sabes donde podrá su figura tu contrincante. Para esto decidimos implementar el algoritmo MINIMAX en el triqui. Con esto garantizamos que, puedas ver cuál es la mejor posibilidad para ganar en cualquier estado del tablero, o como mínimo tendrás un empate. Podrás poner el tablero en el que juegues y te mostraremos cuál es tu mejor chance, y lo que tendrás que hacer paso a paso para lograrlo. Ahora te estamos ofreciendo una ayuda para que satisfagas tus deseos de victoria.

Funcionalidad de la herramienta

La funcionalidad de la herramienta consiste analizar situaciones de partidas de triqui. Para así poder resolverlas en la menor cantidad de jugadas. Para llegar a esto, lo que hicimos es que la herramienta pueda analizar los movimientos y verificar que cumplan estas características; reducir las posibilidades del rival de ganar y realizar movimientos que garanticen ganar o empatar en el menor tiempo posible. Con este fin, utilizamos el algoritmo MINIMAX, este mediante valores numéricos puede decirnos qué jugada maximizará nuestro juego, en otras palabras nuestras posibilidades de ganar; por ejemplo, en un tablero aleatorio hay tres posibilidades, la primera impedir que haya dos figuras contiguas del rival, segunda intentar tener dos figuras seguidas y tercera ponerla en cualquier casilla vacía; lo que haría la herramienta sería, elegir la opción 2 que si logra tener dos seguidas antes que el rival, esto lo obligará a el tener que evitar su inminente derrota, lo que maximizará nuestro abanico de jugadas y por ende las posibilidades de ganar.

Utilizando estas reglas la herramienta puede resolver cualquier situación satisfactoriamente.

Algoritmos y estructuras de datos

Utilizamos el algoritmo de MINIMAX el cual funciona de manera recursiva y dentro de la teoría de juegos es un método para minimizar la pérdida máxima esperada en juegos en los que se tiene un adversario, como ajedrez, conecta 4, damas y triqui.

El algoritmo revisa si la jugada actual es la mejor jugada posible, para esto considera todos los movimientos en donde el valor de la jugada sea aquel que le proporcione mayor posibilidad de ganar, asumiendo que el oponente va a jugar de manera a optima.

Para explicar mejor lo que utiliza MINIMAX como valor de la jugada, asumamos que hay dos jugadas posibles, la jugada A y la jugada B, en las cuales puede ganar en X cantidad de movimientos. En la jugada A gana en X = 2 movimientos y en la jugada B gana en X = 4. Sabiendo que los números positivos son las posibilidades de ganar, supongamos que 10 seria ganar por lo que, tomando nuestro primer enunciado de utilizar la menor de cantidad de movimientos, la jugada A queda 10 – 2 = 8 y B 10 – 4 = 6. Ahora que se ve que el valor de A es mayor, este es el valor que tomara el algoritmo, aunque ambos caminos lo lleven a la victoria.

Se debe hacer la aclaración de que la función MINIMAX implementada por nosotros, hace uso de otras funciones como FindBestMove(FBM) y ValuePlay. FMB encuentra cual es la jugada siguiente que le aportara al algoritmo un mayor valor para llegar a su victoria, hace algo parecido a lo que haría MINIMAX pero solamente con el paso siguiente y no con toda la solución al problema; y ValuePlay nos dice en la jugada que nos encontremos cuál es su valor exacto. Retomando el ejemplo anterior, sabemos que para seguir la jugada A en algún momento debemos saber cuál es camino más optimo a tomar para llegar a un valor de 8, por lo que ValuePlay nos diría los valores de los movimientos posibles y FBM elegirá el mejor camino para llegar al 8.

Para poder implementar este algoritmo y resolver nuestro problema usamos vectores y maps. La clase vector fue utilizada para desarrollar la estructura del tablero, un vector de tamaño nueve el cual cada posición numérica del vector represente una casilla. Si esto se quiere ver de mejor manera, es similar a como esta organizado un teclado numérico, por lo que extrapolando esto se ve que la posición 1 del vector es el 1 en el teclado y la primera casilla del tablero. Luego la librería map, tiene la función de facilitar el trabajo de las funciones FindbestMove, ValuePlay y más importante MINIMAX, esto se logra dándole una relación de valores numéricos a los símbolos de juego que son el vacío (“ “), la cruz(“X”) y el circulo(“O”), de esta manera utilizando 0, 1 y 2 se puede con números enteros acceder a datos de carácter lo cual facilito su optimización de estados tardíos del desarrollo al igual que su orden y legibilidad.