**DOCUMENTO PROYECTO LINJA**

**INTRODUCCIÓN**

La inteligencia artificial es una de las ramas de la informática de más rápido crecimiento en los últimos diez años. El objetivo principal de la inteligencia artificial no es replicar la inteligencia humana, sino estudiar los procesos simbólicos, el razonamiento no algorítmico y la representación del conocimiento [1].

Los temas tratados en este documento involucran dos aspectos, que han provocado la mayor investigación en inteligencia artificial en las últimas dos décadas: Juegos y algoritmos de búsqueda. Para su desarrollo se describirá el proceso de juego entre dos oponentes corriendo en el tablero, las piezas de linja en el juego serán rojas para el jugador que inicia la partida y negras para el oponente.

En la teoría de juegos que plantea Nash no debemos preguntarnos qué haremos después, sino, debemos preguntarnos qué haremos dependiendo y teniendo en cuenta lo que pensamos que pueden hacer los demás y ellos actuaran según lo que piensen que pueda ser nuestro siguiente movimiento [2]. Haciendo un análisis de las líneas anteriores coincidimos con el compañero de trabajo en utilizar el método de búsqueda MaxMin con una Matriz como estructura de datos para el desarrollo del juego Linja.

**OBJETIVO GENERAL:**

Desarrollar una aplicación utilizando el lenguaje de programación Python y métodos algorítmicos de búsqueda para simular el funcionamiento del juego de estrategia Linja aplicando inteligencia artificial.

**OBJETIVOS ESPECÍFICOS:**

* Diseñar una estructura de datos que sea apropiada para el desarrollo del juego.
* Determinar el tipo de búsqueda entre adversarios utilizada en la teoría de juegos.
* Sobreponer conocimientos adquiridos en el área de inteligencia artificial y materias que ayuden a su desarrollo.

**CONCEPTUALIZACIÓN**

**APLICACIÓN TEORÍA DE JUEGOS**

La teoría de juegos estudia las decisiones que tiene un individuo para que tenga éxito de acuerdo con las decisiones que tomen los otros individuos en determinada situación.

Viendo esta definición de la teoría de juegos, podemos definirla como una serie de algoritmos, donde los sujetos, dependiendo del escenario, toman decisiones teniendo en cuenta las elecciones de los demás para poder acertar y llegar a la meta que lo beneficie [3].

Siguiendo esta observación, los juegos aparecen constantemente en la vida cotidiana. Así, la teoría de juegos no es únicamente una manera de adivinar el porte de las personas que participan en un problema, esta es útil también para examinar la justa de precios entre dos tiendas que están en la misma calle, así como para muchas otras situaciones.

La teoría de juegos como despacho numeral no se ha utilizado solamente en los juegos, sino también se ha utilizado en la gestión, estrategia, psicología y también en biología. La teoría del juego presenta los siguientes aportes clave:

* La teoría del juego sirve para comprender situaciones sociales entre personajes en competencia y producir la elección de la decisión óptima de personajes independientes y en competencia en un entorno estratégico.
* Muchos de los escenarios incluyen el dilema del prisionero y el juego del dictador, entre muchos otros.
* Se pueden establecer escenarios del mundo real usando la teoría de juegos, en donde existan situaciones como la competencia de precios, el lanzamiento de productos entre otras para predecir sus resultados.

**FUNIONAMIENTO ARBOL DE JUEGO**

La manera natural de representarlo que puede suceder en un juego es mediante lo que se conoce como árbol de juego, que es un tipo especial de árbol semántico en el que los "nodos" representan configuraciones de tablero y las "ramas" indican cómo una configuración puede transformarse en otra mediante un solo movimiento

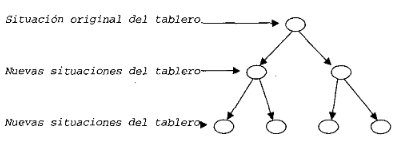


Figura 1. Árbol de Juego [1].

En la Figura1 se hace una representación en forma de árbol de juegos de dos oponentes donde cada vez que se realiza un movimiento estratégico se generan varias situaciones de juego.

**ALGORITMOS DE BUSQUEDA CON ADVERSARIOS**

En la teoría de juegos se encuentran los algoritmos Minimax y Alfa-Beta, en donde, podemos ver la teoría de juegos como una serie de algoritmos, donde los sujetos dependiendo del escenario, toman decisiones teniendo en cuenta las elecciones de los demás para poder acertar y llegar a la meta que lo beneficie. Para el desarrollo del juego Linja se implementará el algoritmo de búsqueda Minimax donde explicaremos su funcionamiento e implementación para el desarrollo del proyecto.

**MINIMAX**

Este algoritmo es el más conocido y utilizado para problemas con exactamente dos competidores, donde hay información perfecta y movimientos alternos, es decir, “después de mí, sigues tú”. Este algoritmo identifica a cada jugador como, jugador MAX y jugador MIN. MAX es el jugador que inicia el juego, para entenderlo mejor, supondremos que MAX somos nosotros, y marcaremos como nuestro objetivo encontrar el conjunto de movimientos que proporcionen nuestra victoria, independientemente de lo que haga el jugador MIN [3].

Pasos del algoritmo Minimax:

* Generación del árbol de juego. Se generarán todos los nodos hasta llegar a un estado terminal.
* Cálculo de los valores de la función de utilidad para cada nodo terminal.
* Calcular el valor de los nodos superiores a partir del valor de los inferiores. Según nivel si es MAX o MIN se elegirán los valores mínimos y máximos representando los movimientos del jugador y del oponente, de ahí el nombre de Minimax.
* Elegir la jugada valorando los valores que han llegado al nivel superior

**REGLAS DE JUEGO LINJA**

Un juego para 2 jugadores a partir de 8 años diseñado por Steffen Mühlhäuser.

**Componentes:** 7 bastoncillos de bambú, 2 x 12 fichas.

**Objetivo del juego**: El objetivo del juego es avanzar las fichas propias lo más cerca posible del otro lado del tablero.

**Preparación del juego**: Cada jugador escoge un color y coloca sus doce fichas tal y como se muestra en la ilustración. El primer jugador se escoge al azar.

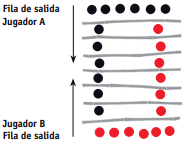


Figura 2. Inicio del juego

**Desarrollo del juego** Un movimiento consiste en hasta tres acciones consecutivas pero separadas. A partir de ahora, les llamaremos “Movimiento Inicial”, “Segundo Movimiento” y “Movimiento Extra”. Los jugadores llevarán a cabo estos movimientos en su turno, siempre en la dirección de la fila de salida del contrario. Solo se permite mover hacia atrás durante el “Movimiento Extra”.

**Movimiento Inicial:** El jugador en su turno mueve cualquiera de sus fichas por encima de un bastoncillo a la siguiente fila.

**Segundo movimiento:** El número de fichas en la fila dónde acaba de saltar una ficha determina la distancia del Segundo Movimiento. Se cuentan tanto tus fichas como las de tu oponente exceptuando la ficha que, con su salto, acaba de entrar en esa fila. El Segundo Movimiento sólo puede ser realizado por una ficha.

**Situaciones especiales:** Si una ficha sale del tablero (la fila de salida del contrario) durante su Segundo Movimiento con puntos sobrantes de movimiento, estos se pierden. Si una ficha mueve a una fila vacía o sale del tablero (la fila de salida del contrario) con su Movimiento Inicial, el jugador no tendrá Segundo Movimiento.

**Movimiento Extra:** Cuando una ficha llega a la fila de salida del contrario con el número exacto de movimientos, dispondrá de un Movimiento Extra. Podrá, inmediatamente, mover una cualquiera de sus fichas una fila hacia adelante o hacia atrás. Con esto terminará su turno.

**Filas ocupadas**: No podrá haber nunca más de seis fichas en una fila. Una fila así ocupada, sin embargo, sigue permitiendo el movimiento a través de ella y deberá contarse al mover. El número de fichas fuera de tablero (fila de salida) no está restringido.

**Fin de la partida:** La partida termina, inmediatamente, cuando las fichas de ambos jugadores se han sobrepasado completamente. Pasaremos entonces a puntuar la partida. Las fichas que aún estén en la zona de puntuación del oponente al final de la partida, contarán como puntos negativos según la fila en que estén.

Ejemplo de puntuación:

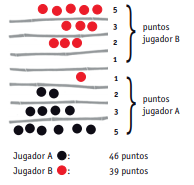


Figura 3. Puntuación final

El jugador con más puntos será el ganador. El perdedor iniciará la siguiente partida.

**PROCEDIMIENTO EXPERIMENTAL:**

Tomando en cuenta las informaciones anteriores es importante mencionar que esta investigación es de tipo experimental, debido a que se desarrollaron pruebas de escritorio de los algoritmos y se determina el algoritmo que mejor se comprende y optimizara de mejor manera los movimientos realizados en Linja.

En el caso del juego que se implementó fue crucial realizar pruebas con el juego, familiarizarse con él, conocer las reglas, técnicas y métodos que se emplean para alcanzar la victoria. De este modo al momento de implementar el juego solo hizo falta tomar en cuenta cada uno de estos conocimientos y llevarlos a ejecución.

**ESTRUCTURA DE DATOS**

Uno de los pasos que se deben realizar en las primeras fases del desarrollo del proyecto es tener claro la estructura de datos en la que se va a trabajar para ello se debe entender el funcionamiento del juego Linja.

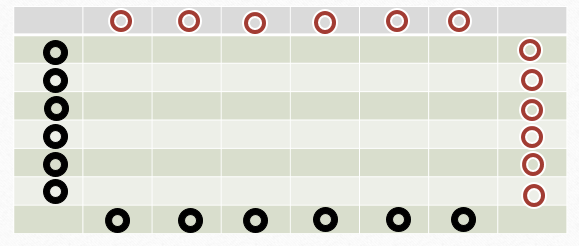


Figura 4. Estructura de datos: Matriz

La matriz tiene una estructura basada en filas y columnas en la cual podemos almacenar datos del mismo tipo (int, float, String etc.) las dimensiones son 8x8 como podemos observar en la figura 4 (8 filas y 8 columnas). La dimensión de la matriz se determinó por el fácil manejo de los datos y dentro de cada espacio matricial el tipo de dato va ser equivalente.

**ESTADO INICIAL**

En el estado inicial del juego tenemos que posicionar las fichas según las reglas del juego como se puede observar en la figura 4, para la ubicación de cada una de ellas dentro de la matriz 8x8 nos guiaremos por la tabla 1.

|  |  |  |  |
| --- | --- | --- | --- |
| **COLOR FICHA** | **N° COLUMNA** | **N° FILA** | **POSICIÓN MATRIZ** |
| ROJO | 0 | 1 | [0,1] |
| ROJO | 0 | 2 | [0,2] |
| ROJO | 0 | 3 | [0,3] |
| ROJO | 0 | 4 | [0,4] |
| ROJO | 0 | 5 | [0,5] |
| ROJO | 0 | 6 | [0,6] |
| ROJO | 1 | 7 | [1,7] |
| ROJO | 2 | 7 | [2,7] |
| ROJO | 3 | 7 | [3,7] |
| ROJO | 4 | 7 | [4,7] |
| ROJO | 5 | 7 | [5,7] |
| ROJO | 6 | 7 | [6,7] |
| NEGRO | 1 | 0 | [1,0] |
| NEGRO | 2 | 0 | [2,0] |
| NEGRO | 3 | 0 | [3,0] |
| NEGRO | 4 | 0 | [4,0] |
| NEGRO | 5 | 0 | [5,0] |
| NEGRO | 6 | 0 | [6,0] |
| NEGRO | 7 | 1 | [7,1] |
| NEGRO | 7 | 2 | [7,2] |
| NEGRO | 7 | 3 | [7,3] |
| NEGRO | 7 | 4 | [7,4] |
| NEGRO | 7 | 5 | [7,5] |
| NEGRO | 7 | 6 | [7,6] |

Tabla 1. Estado inicial matriz

El desarrollo de la matriz que se observa en la tabla 1 dejando las posiciones de las esquinas libres: [0,0], [0,7], [7,0], [7,7] es por una de las reglas de juego en linja que nos indica que el número de fichas que logren llegar a la meta en las columnas no tiene límite, si llegado el caso una de las fichas corona 6 o más fichas el espacio de llegada con dos espacios más en la memoria.

**ESTADO FINAL**

Para que una partida de Linja finalice todas las fichas se tienen que sobrepasar como se observa en la figura 5 y posterior mente realizar las cuentas según la ubicación de cada ficha en la columna que se encuentre

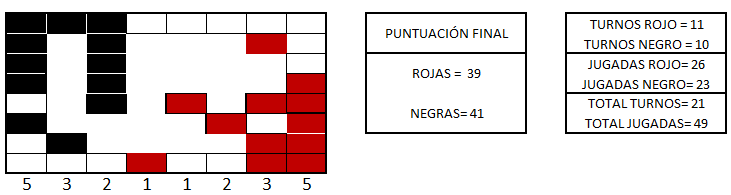


Figura 5. Estado de Finalización

Una vez finalizada la partida (como ejemplo tomamos la figura 5) realizamos el conteo de puntos según la tabla 2, en este caso el ganador es el jugador de fichas negras “Min”. El número de turnos realizado por cada jugador hace referencia a las intervenciones de cada uno y el número de jugadas es el movimiento de una ficha diferente en cada turno.

|  |  |  |
| --- | --- | --- |
| **COLOR FICHA** | **N° COLUMNA** | **PUNTUACIÓN** |
| NEGRO | 0 | 5 |
| NEGRO | 1 | 3 |
| NEGRO | 2 | 2 |
| NEGRO | 3 | 1 |
| ROJO | 4 | 1 |
| ROJO | 5 | 2 |
| ROJO | 6 | 3 |
| ROJO | 7 | 5 |

Tabla 2. Puntuación final

El posicionamiento de la columna dentro de la matriz es el dato referente para asignar la puntuación final como se puede observar en la tabla 2, las filas nos ayudan para el desplazamiento de las fichas y generar un orden en el tablero.

**SIMULACIÓN DE JUEGO**

Para entender cada una de las reglas de juego y saber que movimientos son válidos y cuales no, se realizó una simulación de una partida de Linja para que la implementación se realice de una mejor manera. En la figura 6 se observa un estado inicial, el jugador que empieza la partida es el de las fichas rojas con 2 movimientos realizados para este caso lo llamaremos Max, el jugador de las fichas negras el cual realiza el siguiente turno y dos movimientos se denominara Min.

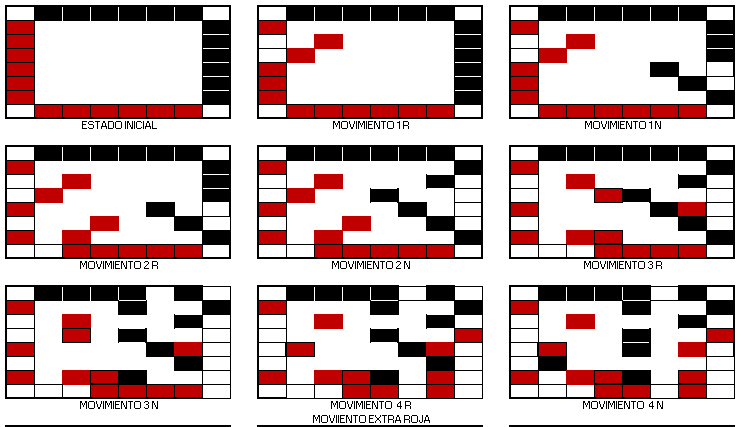


Figura 6. Simulación de juego

El jugador Max empieza su turno que consta de dos movimientos, en el primer movimiento solo puede avanzar una fila hacia la zona del rival, en el segundo movimiento se tiene que contar el número de fichas que se encuentran en la columna antes de ingresar la ficha, el número de fichas determinara la cantidad de filas que puede avanzar.

En el movimiento 4 de las fichas rojas de la figura 6 el jugador realizo 3 movimientos en su turno, el movimiento extra se debe a que una de sus fichas llego a la meta con el número de desplazamientos de filas exacto, si el número de desplazamiento de la fila sobrepasa la meta ya no se realizará el movimiento extra y su turno constará solo de 2 movimientos.

**DISCUSIONES Y RESULTADOS**

Uno de los resultados favorables que tuvo el desarrollo del juego linja es que sea planeo una ingeniería de software correcta al establecer el uso de los métodos de búsqueda y la estructura de datos, estos fueron implementados en Python y el resultado fue el esperado.

En la implementación del método de búsqueda min Max nos fue de gran utilidad con la implementación de métodos recursivos para prever posibles movimientos y desplazar la ficha correctamente según las reglas de juego preestablecidas, pero uno de los fallos encontrados dentro del desarrollo es la cantidad de posibles movimientos generados por tal motivo el programa en ciertos casos revienta para ello se da una explicación lógica mediante la figura 7

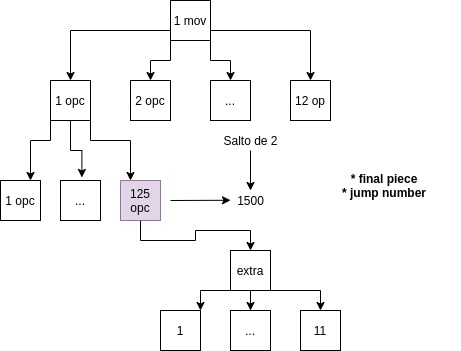


Figura 7. Posibles movimientos

Como se puede observar en la figura 7 se generan gran cantidad de decisiones que se pueden tomar durante el desarrollo del juego, una de las soluciones que se analizó con el grupo de trabajo fue implementar un método de búsqueda que complementara el trabajado para reducir los posibles movimientos, el método que se pensó fue el de poda alfa-beta.

En general el resultado fue satisfactorio, se aplicaron conocimientos adquiridos a lo largo del tiempo que fueron de gran provecho para lograr objetivos planteados y modos de juego que serán de gran interacción para el usuario.

**CONCLUSIONES**

* El algoritmo de Minimax es uno de los más útiles para el desarrollo de juegos entre contrarios, pero una de sus falencias es que ocupa una mayor de memoria
* La implementación de un método de búsqueda recursivo ahorra líneas de código y el esfuerzo del desarrollador.
* El análisis de juegos estratégicos en IC utiliza métodos de búsqueda de adversarios que optimizan su funcionamiento.

**BIBLIOGRAFIA**

[1] Ruiz, L., Motzarenko, A. and Leon, J., 2013. JUEGO MANCALA CON ALGORITMO MINIMAX REALIZADO CON INTERFAZ GRÁFICA EN JAVA. [ebook] Madrid, p.8. Available at: <http://www.it.uc3m.es/~jvillena/irc/practicas/12-13/05mem.pdf> [Accessed 17 March 2021].

[2] N. R. Becerra Correa, “Árboles de Juegos”, Ing. Inv., no. 45, pp. 53-61, Jan. 2000.

[3] J. Castillo, J. Morán, A. Lan, y N. Béliz Osorio, Comparación de los algoritmos de búsqueda en problemas de videojuegos de estrategia, Rev-RIC, vol. 6, n.º 2, pp. 60-66, dic. 2020.