RESÚMEN

En la actualidad encontramos software, juegos o aplicativos que nos permiten ya sea divertirnos, llegar a un destino, hacer una búsqueda, con sólo hablar poner una alarma, entre otros. La inteligencia artificial ha permitido que se puedan hacer este tipo de cosas y en un tiempo no muy lejano otras más.

A causa del proceso en MACC, surge la idea de implementar en un juego una inteligencia artificial, con el fin de que con este conocimiento no sólo se aplique al desarrollo de juegos, sino además para futuro, software que permitan resolver problemas de la cotidianidad.

Para ello nos hemos propuesto desarrollar el juego tic tac toe, implementado con inteligencia artificial Minmax.

De igual manera este proyecto es beneficioso ya que motiva a los estudiantes en el estudio y aplicación de la inteligencia artificial que puede ser soportada por la teoría de juegos, buscando crear una actitud crítica y pensamiento creativo.

Los impulsores de este proyecto son los estudiantes David Santiago Moreno y Edwin Alejandro Forero.

INTRODUCCIÓN

Cuando pensamos en inteligencia artificial, lo asociamos a películas o series en donde las máquinas tienen prácticamente una autonomía y son más inteligentes que el humano, pero la realidad es que aún estamos muy distantes a que eso pueda ocurrir.

En la industria de los videojuegos, esta inteligencia es muy recurrida para el desarrollo de juegos y software que permita crear o facilitar algún proceso. Podemos ver el caso de la computadora DeepBlue que derrotó al campeón mundial de ajedrez Kasparov para el año 1997.

Por otra parte, evidenciamos software capaz de decirnos cuál es la ruta más óptima para llegar a un destino en específico o que nos permita encontrar por medio de un buscador cierta información; es decir, la inteligencia artificial no sólo se aplica para los juegos, sino que además está presente en nuestra cotidianidad.

PROBLEMA

Como parte de nuestro proceso en MACC y proceso de aprendizaje, surge la idea de entender cómo desarrollar un juego implementado con una inteligencia artificial, que, para futuro con los conocimientos adquiridos durante la carrera, permita utilizarlos para resolver problemas del día a día.

Por esta razón nuestro problema es: ¿Cómo diseñar, desarrollar e implementar en el juego tic tac toe la inteligencia artificial Minmax?

OBJETIVO

* Implementar el algoritmo Minmax en el juego tic tac toe (tres en raya)

MARCO REFERENCIAL

* TIC TAE TOE:

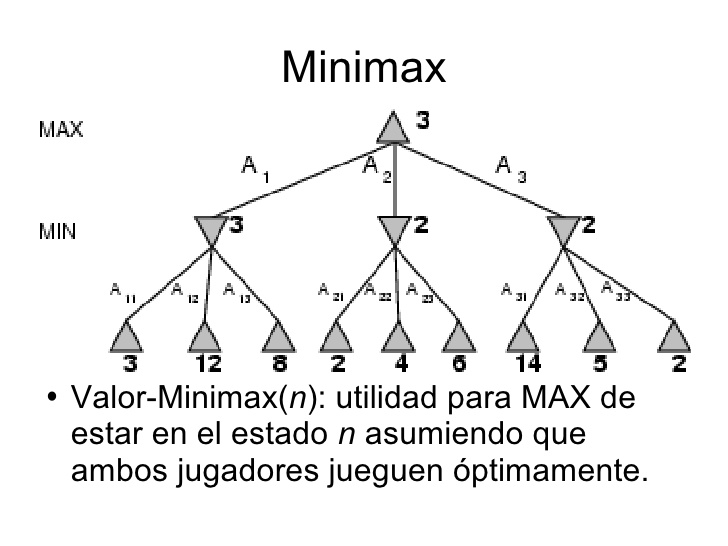
Juego en el que dos jugadores escriben O o X en un patrón de nueve cuadrados. Lo gana el primer jugador que coloca tres Os o tres X en línea recta.

* ALGORITMO MINIMAX:

El algoritmo de minimax consiste en la elección del mejor movimiento para el computador, suponiendo que el contrincante escogerá uno que lo pueda perjudicar, para escoger la mejor opción este algoritmo realiza un árbol de búsqueda con todos los posibles movimientos, luego recorre todo el árbol de soluciones del juego a partir de un estado dado, es decir, según las casillas que ya han sido rellenadas. Por tanto, minimax se ejecutará cada vez que le toque mover a la IA.

Su función de evaluación depende de si ganó, perdió o empató y además cada movimiento es determinado por cada nodo y el valor que le corresponda.

Estos valores se obtienen del nodo inferior en el cual se elige el máximo si le corresponde el turno a la máquina y en mínimo si es turno del jugador.



MARCO METODOLÓGICO

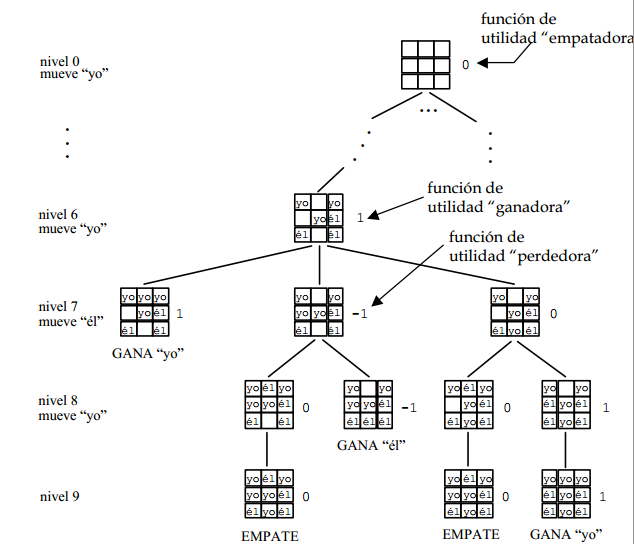
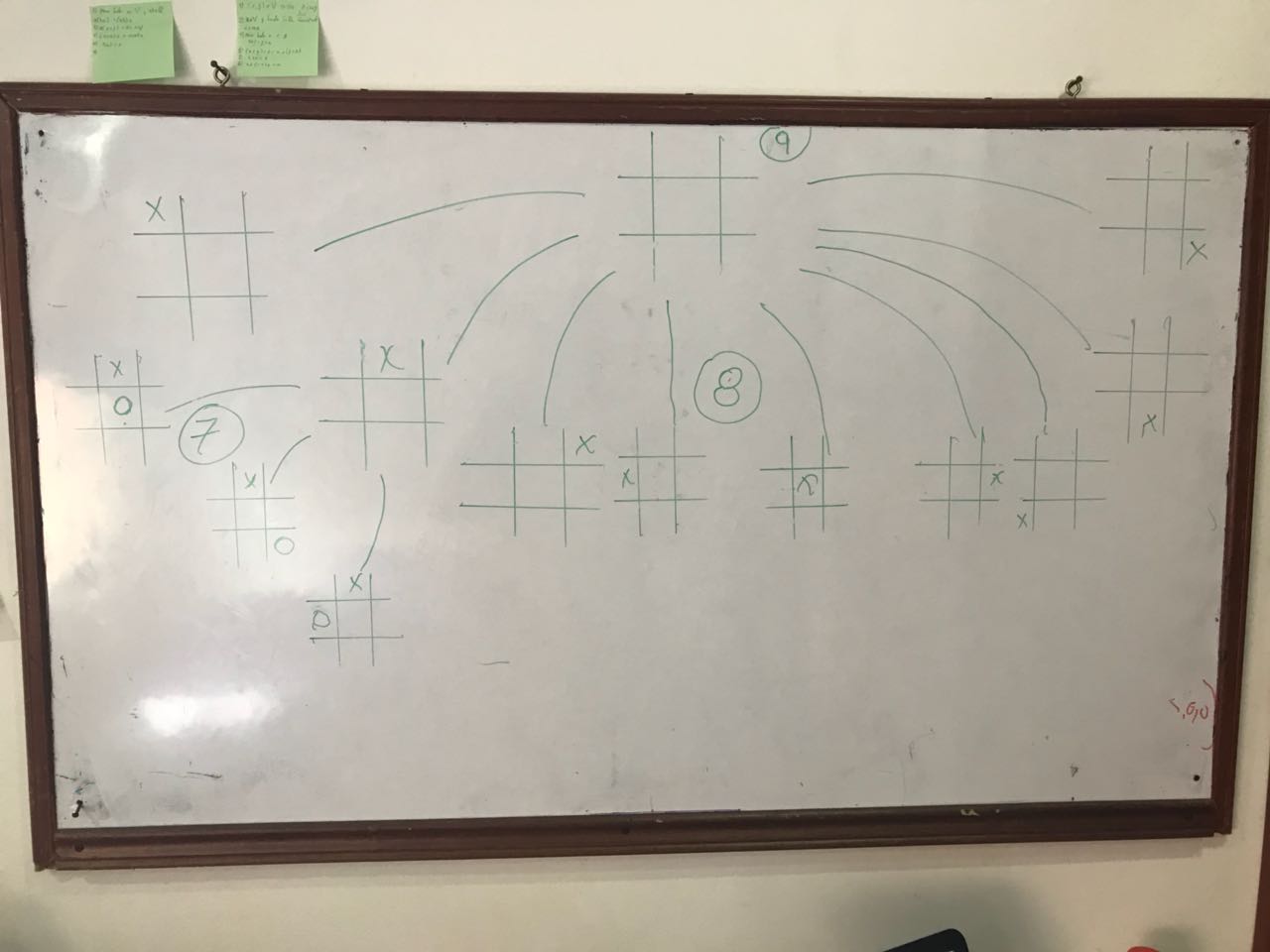
* DISEÑO:

Para el diseño de nuestro proyecto, se tuvo que hacer elección de algún algoritmo que se ajustara al tipo de juego a desarrollar. Finalmente se hizo uso del algoritmo Minmax debido a ciertas características, como lo son:

1. Recursividad y utilidad.
2. Ofrece información completa, es decir, requiere la información de los jugadores en tiempo real.
3. El algoritmo es limitado. En nuestro caso no requerimos de un número infinito de jugadas y por tanto se puede tener un control jugada a jugada.
4. Posee capacidad de aprendizaje.
5. Es muy recurrido para juegos de 2 personas.
6. Se pueden hacer modificaciones para hacerlo más eficiente.

* DESARROLLO

Luego de la elección del algoritmo, se hicieron simulaciones de un juego que fueron representadas por bosquejos y al tiempo un dibujo del árbol de decisión, para entender el funcionamiento del algoritmo.



* IMPLEMENTACIÓN:

Nuestro proyecto fue desarrollado en el lenguaje Python 2.7.

Para implementar el algoritmo, dividimos el código en diferentes funciones, la primera función le pregunta al usuario si quiere ser X o O y decimos que si el jugador elige ser x entonces la maquina tiene que valer O e inversamente.

La siguiente función también pregunta al usuario si quiere comenzar, a lo que responde Si o No (la primera en mayúscula). Si responde que si entonces retornamos un 1 y si juega de segundo retorna un 0.

Como tercera función gana() permite saber el orden de jugadas, así como también saber el estado del juego en este caso evaluar si alguien ganó o si hay empate.

En la siguiente función se implementa parte del algoritmo en donde la función mira las posibles victorias y si encuentra que en esas posiciones hay algo mira la posición. Si ve que es la letra de la persona por ejemplo si el jugador elige ser x entonces retorna 1 y si es maquina retorna 0 y si ve que no hay 3 en línea y ya están ocupadas todas las casillas entonces es empate.

En la función jugadorp\_mueve() pedimos la posición de donde quiere jugar y mientras ese lugar en la lista sea diferente de vacío que vuelva a pedir la posición y si es vacío que retorne la posicion.

Luego, en la función jugadorm\_mueve() si maquina mueve elegimos las mejores jugadas iniciales que son las esquinas y el centro creamos una lista vacía que le vamos agregando elementos de las posibles jugadas que puede hacer y después hacemos un ciclo en las jugadas posibles. Si la funcion gana ==0 entonces la maquina gana y que me retorne las jugadas en las que gana, vaciamos la lista para que lo vuelva a hacer, y ahora miramos si el adversario puede ganar en 1, y si nosotros no podemos ganar en uno entonces devolvemos las jugadas en donde gana el adversario y las ponemos ahí; si no elegimos entre las mejores jugadas y hacemos un random entre esos números.

Y por último nuestra funcion que la llamamos ejecutable(). En esta funcion llamamos las funciones, en donde se va haciendo los diferentes procesos de las anteriores funciones, pero con la diferencia de que en esta instancia se imprime el tablero.