Universidad Autónoma de Nuevo León
Facultad de Ciencias Físico Matemáticas
San Nicolás de los Garza

Tarea 7

Inteligencia artificial

Docente: JUAN PABLO ROSAS BALDAZO

Matrícula	Nombre
1947480	FELIX YAHVEH ALANIS CANDELARIA
1963196	JUAN CARLOS DIAZ GONZALEZ
1962111	JOHANN JOSEPH VELÁZQUEZ ANTONIO

San Nicolás de los Garza, a 24 de abril del 2022

Minimax Tic Tac toe

Equipo:

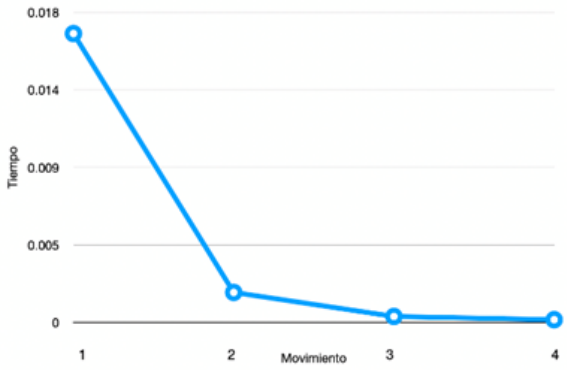
Felix Yahveh Alanis Candelaria	1947480
Juan Carlos Diaz Gonzalez	**1963196**
Johann Joseph Velazquez Antonio	1962111

GITHUB: https://github.com/Johann-28/IA

```
función terminal_test(tablero):
    para cada fila en tablero:
        si fila[0] == fila[1] == fila[2] y fila[0] no es Nulo:
            devolver Verdadero, fila[0]

    para cada columna en rango(3):
        si tablero[0][columna] == tablero[1][columna] == tablero[2][columna] y tablero[0]
[columna] no es Nulo:
            devolver Verdadero, tablero[0][columna]

    si tablero[0][0] == tablero[1][1] == tablero[2][2] y tablero[0][0] no es Nulo:
        devolver Verdadero, tablero[0][0]

    si tablero[0][2] == tablero[1][1] == tablero[2][0] y tablero[0][2] no es Nulo:
        devolver Verdadero, tablero[0][2]

    si todos([todos(fila) para fila en tablero]):
        devolver Verdadero, Nulo

    devolver Falso, Nulo

función utilidad(tablero, jugador):
    is_terminal, ganador = terminal_test(tablero)

    si ganador == jugador:
        devolver 1
    sino si ganador es Nulo:
        devolver 0
    sino:
```

```
        devolver -1

función imprimir_tablero(tablero):
    imprimir '-------------'
    para cada fila en tablero:
        imprimir '|',
        para cada elem en fila:
            símbolo = ' ' si elem es Nulo sino elem
            imprimir ' ' + símbolo + ' |',
        imprimir '\n-------------'


función minimax(tablero, profundidad, jugador):

    si profundidad == 0 o terminal_test(tablero)[0]:
        devolver utilidad(tablero, jugador)

    si jugador == 'X':
        valor = menos infinito
        para cada acción en acciones(tablero):
            valor = max(valor, minimax(resultado(tablero, acción), profundidad-1, 'O'))
        devolver valor

    sino:
        valor = más infinito
        para cada acción en acciones(tablero):
            valor = min(valor, minimax(resultado(tablero, acción), profundidad-1, 'X'))
        devolver valor


función actions(tablero):
    devolver [(i, j) para i en rango(3) para j en rango(3) si tablero[i][j] es Nulo]

función result(tablero, acción):
    i, j = acción
    nuevo_tablero = [fila.copiar() para fila en tablero]
    nuevo_tablero[i][j] = 'X' si terminal_test(tablero)[1] == 'O' sino 'O'
    devolver nuevo_tablero

función get_best_action(tablero, profundidad, jugador):
    # Obtener la mejor acción para el jugador
    mejor_valor = float('-inf')
    mejor_accion = None
    for accion in acciones(tablero):
        # Calcular la utilidad de cada posible acción
        valor = minimax(resultado(tablero, accion), profundidad - 1, jugador)

        # Si el valor es mayor que el mejor valor hasta el momento,
        # actualizar el mejor valor y la mejor acción
        if valor > mejor_valor:
            mejor_valor = valor
            mejor_accion = accion

    return mejor_accion
```

Movimiento (Inputs)	Tiempo promedio jugado por computadora para la respuesta
1	0.0167630195617676
2	0.00174336433410645
3	0.000348758697509766
4	0.000160789489746094



Tabla

Grafica

Entre menos opciones tiene la computadora para colocar su marca, menos se tardará en encontrar una solución, por eso el tiempo disminuye conforme avanzan los turnos	Se realizaron 5 pruebas en donde se probó el tiempo promedio en responder la maquina. Las características del equipo en que se corrió fueron las siguientes: Procesador: AMD Ryzen 5 3.30 GHz