**Práctica 2 – 4 en raya**

# RESUMEN

El trabajo que se analiza a continuación consiste en un proyecto basado en AgentSpeak, utilizando Jason, con el propósito de crear un sistema inteligente que siga una estrategia para ganar o perder, según se le indique, a un juego conocido como “4 en raya”, cuyo tablero ha sido implementado en Java y se utiliza como punto de referencia sobre el que realizar las jugadas.

# INTRODUCCIÓN

En este proyecto se hará uso del entorno, conocido como “tablero”, desarrollado en Java y facilitado por el profesor. Sobre este, el agente “jugador” tendrá que colocar una serie de fichas con el objetivo de conseguir colocar 4 seguidas en línea recta, si la estrategia a ejecutar busca ganar, sin embargo, si la estrategia busca perder, intentará evitar a toda costa colocar 4 fichas en línea recta, ya sea horizontal, vertical o diagonalmente.

El tablero tiene una dimensión de 8x8 cuadrículas, por lo que es posible colocar hasta 64 fichas, existiendo la probabilidad de terminar en empate.

El jugador se enfrentará a otro, utilizando un sistema de juego basado en turnos, durante los cuales se puede colocar única y exclusivamente 1 ficha, por lo que puede llegar a ser complejo lograr el objetivo de ganar o perder debido a las diversas posibilidades que dicho tablero permite.

# DESARROLLO

### Inicio del agente

El jugador inicia por defecto con el plan “start”, el cual obtiene su nombre como agente para así poder saber cuándo es su turno.

Finalmente, cuando le adjudican el turno, procede a ejecutar el plan “jugar”, que comprueba cual es la estrategia a seguir (ganar o perder) para así realizar unas acciones u otras.

### ACCIONES

Para la colocación de una ficha en el tablero, el jugador utiliza put(X,Y), siendo X la posición horizontal e Y la posición vertical, que pueden tener valores entre 0 y 7.

En cuanto a la tolerancia a las trampas, existe un plan por defecto para toda la información que llegue de otras entidades que no sean el tablero o el propio jugador, mostrando un mensaje que indica que ignora a dicha entidad.

### ESTRATEGIAS

#### JUGAR A GANAR

El jugador comprueba si tiene que iniciar la partida, averiguando si están libres todas las posiciones del tablero, solicitándole dicha información al entorno.

En caso afirmativo, coloca en la posición (4,3), que es una de las 4 posiciones más céntricas del tablero.

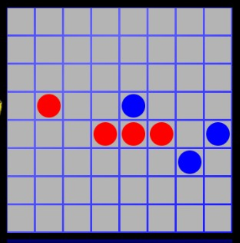
Para cualquier otro caso lo primero que hace es comprobar que no puede hacer 4 en raya, es decir, que no puede ganar en este turno, y que el jugador contrario no tenga forma de hacer 4 en raya en su siguiente turno.

Para comprobar si puede hacer el 4 en raya, llama a cuatroEnRaya(BestX.BestY), el cual buscará si se cumple que hay 3 en raya desde la posición de cada ficha aliada y la cuarta posición en línea esté libre para colocar una ficha. En caso afirmativo, almacena en BestX la posición horizontal de dicho hueco y en BestY la posición vertical.

En caso de no llegar a encontrar un posible 4 en raya, el predicado será falso, por lo que no se cumple.

A continuación, comprueba si el jugador contrario puede hacer 4 en raya empleando un método similar llamado bloquearCuatroEnRaya(BestX,BestY), el cual busca 3 fichas alineadas enemigas y si la cuarta posición en esa línea está libre, almacena la posición horizontal y vertical para colocar en ella.

Si esta condición se cumple (el primer predicado devuelve falso y este último se cumple), ya no continúa evaluando otras posibilidades y almacena en NextX y en NextY los valores de BestX y BestY calculados, respectivamente, de manera que el jugador coloca en la posición determinada por ambos parámetros.

Si no se cumple esa primera condición, evalúa la siguiente, que consiste en bloquear al jugador enemigo si en su siguiente turno puede colocar 3 fichas en línea.

Esto se debe a que, si puede hacerlo, puede llegar a provocar que le queden dos posiciones que le otorguen la victoria simultáneamente, como podemos observar en la Figura 1 para el jugador rojo.

Figura 1.

Las posteriores condiciones a evaluar se basan en buscar la mejor opción para constituir la línea de fichas aliadas más larga hasta el momento mediante tresEnRaya(NextX,NextY), dosEnRaya(NextX,NextY), unoEnRaya(NextX,NextY), Se ejecutan en ese orden hasta que alguno de ellos se cumpla.

Todos ellos realizan lo mismo que el predicado cuatroEnRaya(NextX,NextY) explicado anteriormente, pero buscando una línea con un menor número de fichas aliadas.

#### JUGAR A PERDER

La estrategia a seguir para perder es sencilla a diferencia de la anterior ya que se basa en colocar en posiciones aleatorias procurando evitar colocar 4 en raya.

# CONCLUSIÓN

Del trabajo aquí expuesto, es relevante mencionar que la capacidad del jugador, el cual tiene una estrategia a seguir implementada, permite detectar cuando el enemigo tiene posibilidades de crear una situación favorable para él, al mismo tiempo que analiza cual podría ser la mejor jugada favorable para sí mismo, priorizando debidamente para así alcanzar su objetivo (ganar o perder).

Al ser un juego con unas normas, es posible realizar trampas, así que el jugador también tiene unas directrices a seguir para cuando detecta una posible trampa, procurando trabajar exclusivamente con la información que le proporciona el tablero o él mismo.

También es importante mencionar posibles mejoras, ya que existen métodos como “Minimax”, que permiten minimizar la posibilidad de perder, de una manera mucho más recursiva y efectiva que la que nos encontramos en este trabajo.

# FUENTES

1. González Moreno, J.C. (2017, January). Introducción a los sistemas inteligentes (Sistemas Inteligentes vs. Inteligencia Artificial).

Paper presented at the first session of Intelligent Systems subject, Ourense, ESEI.

1. Página oficial de Jason. http://jason.sourceforge.net/wp/
2. Programming multi-agent systems in AgentSpeak using Jason.

Wiley series in agent technology. Rafael H. Bordini, Jomi Fred Hübner and Michael Wooldridge.