DESCONECTA-4

Jorge Soler Padial

1. Análisis del problema

Se nos plantea desarrollar un agente deliberativo para resolver un juego, el "desconecta-4", juego en el que tenemos información perfecta del estado en el que estamos, los estados a los que podemos llegar y los estados posibles para nuestro contrincante.

Se nos plantea hacer un agente capaz de jugar lo mejor posible y, ya que tenemos información perfecta del juego, podemos usar cualquier algoritmo de juegos como el minimax o poda-AlfaBeta para elegir cual de todas las jugadas posibles del árbol es la mejor y llevarla acabo. El problema es que el factor de ramificación del árbol es 8, ya que cada jugada tiene un máximo de acciones posibles y, aunque el factor de ramificación no es muy alto, comparado con el ajedrez (que suele ser 35 o 250 en el go) no podemos llegar hasta los nodos finales para saber cual es la jugada que nos hará ganar cien por cien seguro. Para ello necesitamos especificar una profundidad asequible y una función heurística de evaluación para dar una valoración a los nodos no-terminales del juego, y poder decidir cual es mas conveniente.

2. Solución planteada

Lo primero a elegir sera el algoritmo que usaremos para explorar el árbol de jugadas, yo me decidí por la poda-AlfaBeta que tiene una mayor profundidad 8 contra los 6 de minimax. Las peculiaridades de mi poda-AlfaBeta son que el primer nodo que explora es la acción de explotar la bomba, que es el nodo 7, ya que normalmente las partidas se ganan explotando una bomba así evitamos que escoja ganar por otra rama mas larga y pode la que de verdad nos lleva a ganar instantáneamente, después se expanden los demás nodos en orden del 0 al 6.Lo segundo destacable del mi poda-AlfaBeta es que a la hora de generar los nodos hijos lo hace de uno en uno usando la función GenerateNextMove() en vez de GenerateAllMoves() y así ahorrarse generar todos los nodos en caso de que se de la condición de poda.

Lo segundo a definir es la función heurística que dará valoración a los nodos no-terminales que se tengan que evaluar, mi función recorre cada casilla del tablero comprobando para esa casilla, en caso de que sea del jugador indicado, cuantas posibilidades tiene de hacer 4 en raya. Para ello recorre vertical,horizontal y diagonalmente comprobando que las fichas que tenga en esa dirección sean del mismo jugador o estén vacías, si hay 4 o mas posiciones que cumplan ese requisito se devuelve 1 (horizontalmente puede ser 2) indicando que es posible hacer 4 en raya en esa dirección. Esto se hace para nuestro jugador y para el contrario, también se cuentan cuantas fichas tiene cada jugador, finalmente las posibilidades de hacer 4 en raya del contrario, se les restan las nuestras y se suma con la resta entre el numero de fichas del contrario menos las nuestras. Quedando finalmente:

(4-raya-posibles-Contrario - 4-raya-posibles-mios) + (num-fichas-contrario – num-fichas-mias). Esa sera la valoración que se le de al nodo.

Con esta implementación se le consigue ganar al jugador Ninja en 11 jugadas empezando nosotros y en 7 empezando él.