Autores:
Alex Carballo Henriques
Luis Francisco Blanco Cadavid

Práctica Final Sistemas Inteligentes GR-3

Tabla de contenido

Autores:
Alex Carballo Henriques
Luis Francisco Blanco Cadavid

Autores:
Alex Carballo Henriques
Luis Francisco Blanco Cadavid

Enunciado:

- Jugador (blancas o negras).- Tratará de colocar el mayor número posible de reinas (hasta 2N) en el tablero sin que se ataquen entre sí, ni sean atacadas por otras reinas.
- Configurador.- Su misión será modificar el tablero mediante la colocación de dos tipos de celdas especiales: muros y agujeros
  Ambas celdas impiden que se coloque encima una reina. Los muros permiten crear refugios en el ataque de reinas, los agujeros no impiden el ataque. El configurador podrá poner como máximo N piezas nuevas en el tablero entre bloques y agujeros.
- Los jugadores que juegan con blancas o negras ganan si se dan las mismas circunstancias que en las prácticas anteriores
- El configurador no puede ganar la partida, solo decidirla. Para ello mantendrá una comunicación constante con los jugadores e intentará colocar sus fichas en las posiciones que acuerde con ellos.
- El configurador puede llegar a acuerdos con un jugador diferente en cada turno hasta que agote sus movimientos.
- El configurador llegará a un acuerdo con el jugador que le ofrezca la jugada que permita maximizar una función de beneficio que calcule la razon entre las posiciones ganadoras para negras y las posiciones ganadoras para blancas en la siguiente jugada después de haber escogido la propuesta de blancas o de negras. Si no consigue determinar que existe una posición ganadora para ninguna de las propuestas realizará un movimiento de colocar un bloque en una de las celdas no atacadas del tablero.
- Una posición es ganadora cuando al escoger esa celda en la siguiente jugada, el jugador que la realizó tiene una estrategia ganadora (existe una combinación de colocación de reinas sin que se añadan más bloques o agujeros) que le permite ganar la partida.
- Para realizar la comunicación los agentes utilizarán los siguientes mensajes:
  - .send(configurer, tell, block(X,Y)),, X es la fila e Y la columna de la posición del tablero en que se quiere poner el bloque
  - .send(configurer, tell, hole(X,Y)),, X es la fila e Y la columna de la posición del tablero en que se quiere poner el agujero
  - .send(Ag, tell, accept),, Ag = {white, black}
  - .send(Ag, tell, decline),, Ag = {white, black}

Autores:
Alex Carballo Henriques
Luis Francisco Blanco Cadavid

Cómo se ha planteado el problema:

A partir del entorno dado, se ha generado un agente jugador que puede interactuar tanto con fichas blancas como negras o de configurador del juego.

Rol Jugador:

El jugador posee en su base de conocimientos una lista de posiciones libres que se actualiza con cada movimiento del tablero. Para colocar una reina en su turno busca la primera posición libre y no amenazada del tablero. Para la colocación de bloques, el jugador va a colocar bloques solo si el jugador va perdiendo (tiene menos piezas sobre la mesa) con la intención de maximizar el número de posiciones libres y tener alguna opción para ganar. Para colocar holes lo hace si va ganando, es decir, si tiene más piezas que su contrincante, colocará una un agujero en una posición libre del tablero para quitarle una oportunidad a su adversario.

Rol Configurador:

El agente configurador funciona como un modulador del juego que recibe peticiones de los demás jugadores en el tablero. Las peticiones que llegan a este son el de colocar agujeros para eliminar posiciones libres y el de colocar bloques para liberar posiciones amenazadas por reinas en juego. Para controlar cuales son las opciones más ventajosas para cada jugador, en cuanto a la colocación de bloques, se mide cuantas posiciones libres genera el supuesto bloque. Si es mayor que el que genera el de su contrincante, este colocará el bloque y lo mismo para el otro jugador en caso contrario. Si la colocación es en la misma posición que su contrincante o no se liberan posiciones, no se colocará el bloque en ningún lugar. En cuanto a los agujeros, se da mayor ventaja al jugador que va ganando, ya que bloquea más opciones que el jugador que va perdiendo.

Entorno de Juego:

Se ha limitado el problema de las colisiones a generar una serie de percepciones con las posiciones libres en cada movimiento de los jugadores. De esta forma basta con buscar una posición libre para situar una reina sin peligro de ser atacada por otra de un rival. Así se genera inicialmente todas las posiciones del tablero como libres y a continuación se van eliminando todas las ocupadas y amenazadas. Esto ocurre tanto en el jugador como en el colocador.

Autores:
Alex Carballo Henriques
Luis Francisco Blanco Cadavid

Explicación de planes y creencias

Rol Jugador:

!start: En este plan se define la primer acción de cada turno del jugador. En primer lugar se predisponen como libres todas las posiciones del tablero. Si tiene el primer turno coloca la primera reina en la mitad del tablero con la intención de atacar las máximas posiciones posibles, limitando los oponentes del adversario. En caso de jugar como negras, intenta colocar en la primera posición libre.

!generarLibres: Hace visibles todas las posiciones del tablero como si fuesen libres para el jugador.

!eliminarLibres: Cada vez que se coloca una reina se eliminan de las percepciones de cada jugador las posiciones libres que han sido atacadas o ocupadas.

!eliminarLibresB: Cada vez que se coloca un bloque se eliminan de las percepciones de cada jugador las posiciones libres que han sido liberadas o ocupadas.

! colocarReina: El jugador intenta colocar una reina en una posición libre. Va recorriendo el tablero de arriba a abajo y de izquierda a derecha, hasta encontrar una posición libre.

!colocarHole: Busca huecos sobre posiciones libres del tablero para situar holes si lo considera necesario, inutilizando a las mismas.

!enviarHole: El plan manda la petición al configurador para poner un hole.

Rol Configurador:

!start: Plan inicial del configurador. Inicializa el tablero Genera las posiciones libres del tablero.

!eliminarLibres: Cada vez que se coloca una reina se eliminan de las percepciones de cada jugador las posiciones libres que han sido atacadas o ocupadas.

!eliminarLibresB: Cada vez que se coloca un bloque se eliminan de las percepciones de cada jugador las posiciones libres que han sido liberadas o ocupadas.

!ponerHole: Ante la llegada de una petición calcula cual maximiza las oportunidades de ganar la partida. Se tiene en cuenta que si el número de fichas dl jugador es mayor que el de su contrincante, el primero tiene preferencia a colocar un agujero para disminuir las oportunidades del contrincante.

!comprobar: Ante una llamada de un jugador para colocar un bloque en el tablero calcula que posición de los dos jugadores genera más huecos libres en cada turno. Si se iguala el factor, el Configurador no realizará ninguna acción. Para contar las posiciones libres que generaría colocar un bloque en el futuro se utilizan los siguientes planes: !testLibres, !testLibresA, !testLibresB.

!ponerBloque: Se utiliza este plan para poner el bloque en la posición que señala la estrategia ganadora dentro del plan "!comprobar".

**(Opcional)!colocarBloqueEnLibre: Este plan está comentado. Si ninguna de las estrategias de los jugadores es aceptada por el Configurador, se coloca el bloque en una posición libre del tablero.

!testLibres: Cuenta las posiciones libres que se producen si se coloca el bloque que el jugador ha solicitado. Se añade el percept +libreAux(X,Y). Este percept solo es utilizado por el configurador para esta situación.

!testLibresA: Es un plan auxiliar para eliminar las libresAux que generan las reinas del tabero.

!testLibresB: Es un plan auxiliar para eliminar las libresAux que generan los bloques del tabero.

Beliefs:

queenBlancas(0).	// Número de reinas Negras actuales.
queenNegras(0).	//Número de reinas Negras actuales.
bloquesActuales(0).	// Número de bloques colocados.
agujerosActuales(0).	// Número de agujeros colocados.
v1(0).	//Var. Aux.
v2(0).	//Var. Aux.
holeX(0,0).	//Var. Aux.
holeBlancas(0,0).	//Var. Aux, Futura posición solicitada de un Hole blancas.
holeNegras(0,0).	//Var. Aux, Futura posición solicitada de un Hole negras.
libresBlancas(0,0,0).	//Var. Aux, Futuras posiciones solicitadas de un Bloque blancas
libresNegras(0,0,0).	//Var. Aux, Futura posición solicitada de un Bloque negras.