UNIVERSIDAD DE GRANADA

Inteligencia Artificial

Práctica 2
Agentes Reactivos/Deliberativos
(Los extraños mundos de BelKan)

Carrasco Chicharro, David
15520228-N

1. Introducción

Para esta práctica me planteé resolver el comportamiento deliberativo del agente con un algoritmo DFS (búsqueda en profundidad), pero al implementarlo y ejecutarlo se producían muchos errores difíciles de subsanar, por lo que tomé la decisión de utilizar un algoritmo BFS (búsqueda en anchura), pues podía partir del algoritmo previamente implementado para modificarlo y además este algoritmo encontraba siempre la solución óptima al problema.

2. Implementación

2.1. Agente deliberativo

La búsqueda del camino hasta el objetivo parte de la variable `st`, que contiene los datos del origen. Para la implementación he utilizado el método visto en las clases de prácticas, utilizando una lista de nodos abiertos y otra de nodos cerrados. Se realiza una búsqueda en anchura en la que la lista de nodos abiertos, `visitado`, ha sido implementada con un vector de pares de enteros en los que se almacenan las filas y columnas de cada casilla visitada en la búsqueda, y que no volverá a visitarse más para ahorrar tiempo y memoria; por otra parte, la lista de nodos cerrados es una cola, `Q`, de pares de estados en la que se almacena cada nodo pendiente de visitar junto con su padre. Además en un vector de pares de estados (`padre`) se almacena el camino a seguir por el agente una vez encontrado el objetivo.

Una vez encontrado el camino óptimo se crea el plan de acciones siguiendo las casillas por las que debe pasar el agente, almacenadas en el vector `padre`, guardando cada acción en la lista de acciones (`plan`) según la dirección que toma el agente y su orientación en cada momento.

2.2. Agente reactivo

Para el nivel 2 se hace uso del agente reactivo, que tiene que tener en cuenta cuándo un aldeano se interpone en su camino. En ese caso se hace uso de la función `pathAldeano()`, que tomará las siguientes decisiones:

Si el agente tiene pensado seguir recto y hay un aldeano en su camino, en caso de no tener obstáculos que se lo impidan, lo bordeará por la izquierda o por la derecha, según sea lo más conveniente. En caso de que el agente tuviera previsto girar de todos modos, la acción previamente descrita no se llevará a cabo y el agente girará de acuerdo a su plan preestablecido.

Sin embargo si el agente se encuentra de repente con un aldeano o no tiene opción de bordearlo, éste girará hacia la izquierda o la derecha, según determine, para rehacer un nuevo plan que le permita llegar a su objetivo, siempre y cuando no tuviera previamente planeado girar, en cuyo caso seguirá con su plan anterior.

En caso de que el agente no pueda llevar a cabo ninguna de las acciones previamente descritas, retrocederá un par de casillas hacia atrás y rehará un nuevo plan que le permita conseguir su objetivo.

3. Término

Las acciones a realizar se determinarán en el método `think()`, donde se tendrán en cuenta las variables booleanas `hayPlan` y `accionReactiva` para determinar si el agente debe crear un plan deliberativo, tomar ciertas decisiones en base a acciones reactivas, cambiar un plan, etc., cuyas implementaciones ya han sido descritas en el apartado anterior.