UNIVERSIDAD DE CÓRDOBA

Dpto. Informática y Análisis Numérico

GRADO DE INGENIERÍA INFORMÁTICA

SISTEMAS INTELIGENTES

WILLY EN EL DESIERTO

Sistema inteligente programado en lenguaje CLIPS, que resuelve el juego "Willy en el desierto", basado en "Hunt the Wumpus".

Autor/es Javier Corbacho López
(i52coloj@uco.es)
Eduardo Almeda Luna
(i52allue@uco.es)

Fecha 12/05/2017

Profesor Manuel Jesús Marín Jiménez

Módulos

- **Reconocimiento.** Comprende tanto reglas para disparar la flecha (cuando se sepa dónde está), como las reglas para insertar casillas no-visitadas y casillas de posible peligro.
- **Inconsistencias.** Elimina hechos que podrían producir errores en otros módulos.
- **AfirmarPeligro.** Localiza peligros (arena movediza o serpiente) en el mapa.
- **MoverANoVisitada.** Conjunto de reglas para mover a Willy por casillas no visitadas.
- **IniciarBusqueda.** Regla para, en caso de Willy no tener ninguna casilla no visitada adyacente, iniciar una búsqueda en anchura y encontrar la casilla no visitada más cercana, o en caso de haber realizado ya la busqueda, seguir el camino producido.
- **calcularIr-a.** Módulo utilizado por IniciarBusqueda para encontrar la casilla no visitada más cercana, o para encontrar un camino hacia la serpiente.
- **AlgoritmoBusqueda.** Módulo utilizado por IniciarBusqueda para desarrollar una cola con las casillas adyacentes a la actual.
- **AlgoritmoBusqueda2.** Módulo utilizado por IniciarBusqueda que produce una cola con las casillas que deberá recorrer Willy para llegar a la casilla objetivo.
- **SiguienteMovDeBusqueda.** Módulo utilizado por IniciarBusqueda para conocer la próxima dirección que debe realizar Willy para seguir el camino que ha producido la búsqueda en anchura.
- **Calculos.** Conjunto de reglas para actualizar la posición actual de Willy, las casillas visitadas, y parar a Willy en caso de que llegue al límite de movimientos.
- **Fin_ciclos.** Comprende una única regla para volver a iniciar todo el ciclo.

Tipos de hechos

Tenemos los siguientes tipos de casilla:

- **(no-visitado ?x ?y)**, donde ?x representa la coordenada de abscisas, y la ?y, la ordenada. Representa una casilla adyacente a una visitada y que sabemos que es segura.
- **(visitado ?x ?y)**, donde ?x representa la coordenada de abscisas, y la ?y, la ordenada. Representa una casilla en la que Willy ya ha estado.
- **(posActual ?x ?y)**, donde ?x representa la coordenada de abscisas, y la ?y, la ordenada. Representa la posición actual de Willy en el tablero.
- **(posible-serpiente ?x ?y ?x2 ?y2)**, donde ?x e ?y representan las coordenadas de abscisas y ordenadas, respectivamente, donde puede estar la serpiente. ?x2 e ?y2 son las coordenadas de abscisas y ordenadas, respectivamente, de donde se ha percibido el sonido de la serpiente
- **(posible-temblor ?x ?y ?x2 ?y2)**, donde ?x e ?y representan las coordenadas de abscisas y ordenadas, respectivamente, donde puede estar una arena movediza. ?x2 e ?y2 son las coordenadas de abscisas y ordenadas, respectivamente, de donde se ha percibido el temblor.
- **(seguro-serpiente ?x ?y)**, donde ?x e ?y representan las coordenadas de abscisas y ordenadas, respectivamente, donde se encuentra seguro la serpiente.

- **(seguro-temblor ?x ?y)**, donde ?x e ?y representan las coordenadas de abscisas y ordenadas, respectivamente, donde se encuentra seguro una arena movediza.

Respecto al algoritmo, utilizamos los siguientes hechos:

- **(ir-a ?a ?b)**, donde ?x representa la coordenada de abscisas, y la ?y, la ordenada. Representa la posición objetivo de algoritmo de búsqueda en anchura.
- **(cadenahijos ?x ?y ?x2 ?y2 $?)**, donde cada conjunto de coordenadas representa una casilla inspeccionada por el algoritmo de búsqueda.
- **(cadena-mov ?x ?y ?x2 ?y2 $?)**, utilizado en el algoritmo de búsqueda ,que representa el conjunto de casillas que deberá seguir Willy hasta llegar a la casilla objetivo.
- **(iteracion ?x ?y ?x2 ?y2)**, donde x e y representan una coordenada inspeccionada por el algoritmo, y x2 e y2 son la casilla "padre" de x e y.

Por último, se utilizan los siguientes hechos por otros motivos diferentes:

- **(dirección ?valor)**, que representa la dirección que ha tomado Willy. Sirve para la actualización de la posición de Willy.
- **(movimiento realizado)**, se inserta cada vez que Willy se mueva, para evitar que se pueda mover dos veces en un mismo ciclo (se debe actualizar la posición de Willy primero).

Estrategia seguida

Primero Willy realiza un reconocimiento de su alrededor, para insertar en la base de hechos si las casillas de su alrededor son seguras (no-visitado) o pueden ser peligrosas.

Después revisa la base de hechos para comprobar si puede localizar algún peligro en el mapa.

Respecto a los movimientos, damos preferencia a las casillas no visitadas, la cuales sabemos que son seguras, ya que no hemos detectado ningún sonido o temblor en una casilla contigua.

En el momento en que se Willy se encierre en casillas visitadas, se pone en marcha un algoritmo de búsqueda en anchura para obtener un camino hacia una casilla no visitada. Funciona de la siguiente manera:

Paso 1: se busca una casilla no visitada, y se aserta un hecho ir-a con las coordenadas de dicha casilla no visitada.

Paso 2: se crea una cola en la que se van almacenando, por la derecha, las casillas que se van recorriendo en la búsqueda en anchura.

Paso 3: se evalúa la casilla que aparece más a la izquierda, y se escriben los hijos (las casillas contiguas a la evaluada) en la cola. Se aserta, además, los hechos iteración, los cuales guardan tanto los hijos, como el padre que ha generado dicho hijo. Si ya tenemos una iteración creada, no se asertará otra con las mismas coordenadas.

Paso 4: Se elimina el padre de la cola, y se repiten los pasos 2, 3 y 4 hasta que se encuentra la casilla no visitada con las mismas coordenadas que ir-a.

Paso 5: Cuando se encuentre la casilla no visitada a la que Willy quiere ir, se utilizarán los hechos iteración (obteniendo el padre de cualquier casilla), para obtener un camino desde nuestra casilla actual hasta el objetivo (ir-a).

Paso 6: Willy recorre la cadena de movimientos desarrollada en el paso anterior, comparando cada coordenada obtenida de la cadena con su posición actual para conocer la dirección que tiene que seguir.

Paso 7: al terminar de recorrer la cadena (es decir, la posición actual de Willy coincide con las coordenadas del hecho ir-a), se borran todos los hechos iteración, ir-a y la cadena de movimientos.

Por último, se actualiza la posición actual de Willy y se repite el proceso.

Disparar

Solo se disparará la flecha cuando se tenga asegurada la serpiente, y Willy se encuentre en la misma columna, o en la misma fila que ella.

Inconsistencias

En el código de Willy, se utilizan una serie de reglas para eliminar posibles inconsistencias en la base de hechos. Dichas reglas se explicarán de forma abstracta.

- **inconsistenciaVisitadoSound**: Si una casilla es segura (visitada o no) y está marcada como posible-serpiente, se elimina que sea peligrosa.
- **inconsistenciaVisitadoTemblor**: Si una casilla es segura (visitada o no) y está marcada como posible-temblor, se elimina que sea peligrosa.
- **inconsistenciaPeligros2**: si la casilla al este de la actual está marcada como posible-temblor, pero no como posible o seguro-serpiente, y se percibe Sound, pero no Tremor, entonces esa casilla no tendrá peligro, por lo que se eliminará el peligro y se marcará como no visitada.
- **inconsistenciaPeligros3**: si la casilla al este de la actual está marcada como posible-serpiente, pero no como posible o seguro-temblor, y se percibe Tremor, pero no Sound, entonces esa casilla no tendrá peligro, por lo que se eliminará el peligro y se marcará como no visitada.
- **inconsistenciaPeligros4**: si la casilla al oeste de la actual está marcada como posible-temblor, pero no como posible o seguro-serpiente, y se percibe Sound, pero no Tremor, entonces esa casilla no tendrá peligro, por lo que se eliminará el peligro y se marcará como no visitada.
- **inconsistenciaPeligros5**: si la casilla al oeste de la actual está marcada como posible-serpiente, pero no como posible o seguro-temblor, y se percibe Tremor, pero no Sound, entonces esa casilla no tendrá peligro, por lo que se eliminará el peligro y se marcará como no visitada.
- **inconsistenciaPeligros6**: si la casilla al norte de la actual está marcada como posible-temblor, pero no como posible o seguro-serpiente, y se percibe Sound, pero no Tremor,

entonces esa casilla no tendrá peligro, por lo que se eliminará el peligro y se marcará como no visitada.

- **inconsistenciaPeligros7**: si la casilla al norte de la actual está marcada como posible-serpiente, pero no como posible o seguro-temblor, y se percibe Tremor, pero no Sound, entonces esa casilla no tendrá peligro, por lo que se eliminará el peligro y se marcará como no visitada.
- **inconsistenciaPeligros8**: si la casilla al sur de la actual está marcada como posible-temblor, pero no como posible o seguro-serpiente, y se percibe Sound, pero no Tremor, entonces esa casilla no tendrá peligro, por lo que se eliminará el peligro y se marcará como no visitada.
- **inconsistenciaPeligros9**: si la casilla al sur de la actual está marcada como posible-serpiente, pero no como posible o seguro-temblor, y se percibe Tremor, pero no Sound, entonces esa casilla no tendrá peligro, por lo que se eliminará el peligro y se marcará como no visitada.
- **inconsistenciaPeligro10**: si una casilla está marcada como seguro-serpiente, se deben borrar todas las casillas marcadas como posible-serpiente.
- **inconsistenciaPeligro11**: si una casilla está marcada como seguro-temblor y como posible-serpiente, se debe eliminar el posible-serpiente.

Asegurar la serpiente

- **AsegurarSerpiente1**: cuando hay dos detecciones en vertical.

- **AsegurarSerpiente2**: cuando hay dos detecciones en horizontal.

- **AsegurarSerpiente3**: cuando hay dos detecciones como en la imagen y se tiene como segura la esquina superior izquierda de la imagen

- **AsegurarSerpiente4**: cuando hay dos detecciones como en la imagen y se tiene como segura la esquina inferior derecha de la imagen.

- **AsegurarSerpiente5**: cuando hay dos detecciones como en la imagen y se tiene como segura la esquina superior derecha de la imagen.

- **AsegurarSerpiente6**: cuando hay dos detecciones como en la imagen y se tiene como segura la esquina inferior izquierda de la imagen.

- **AsegurarSerpiente7**: cuando hay una casilla marcada como posible-serpiente, y no hay otra casilla posible-serpiente distinta a la anterior, aunque pueden detectarse en casillas distintas.

Cada vez que se asegura la serpiente, se llama al módulo de inconsistencias.

Asegurar arenas movedizas

Como puede haber varias arenas movedizas alrededor de un temblor, no podemos realizar las mismas comprobaciones que con la serpiente. Solo la podremos asegurar cuando tengamos un posible-temblor detectado en una posición.

Cada vez que se asegura un temblor se llama al módulo de Inconsistencias.

Posibles finales de Willy

En caso de que Willy no encuentre el oasis, se detendrá cuando:

- Cuando lleva 999 pasos.
- Cuando ya no tiene más casillas por visitar, es decir, cuando está rodeado de peligros y no ha detectado a la serpiente (o la ha matado).

Si la serpiente está asegurada, se irá a por ella, para matarla y poder seguir por donde se encontraba, que se marcará como no visitada.