Comisión

Iñaki Garay

Montenegro Emiliano

Consideraciones del proyecto:

El proyecto entregado en esta etapa consta de 5 módulos principales:

* decide\_action.pl
* update\_state.pl
* auxiliar\_predicates.pl
* extras\_for\_agents.pl
* agent\_emig.pl

Todos los módulos se cargan en paralelo cuando se ejecuta el agente pero se decidió dividir la funcionalidad para mantener el grado de significancia y trazabilidad del código y utilizar los predicados propuestos por uno como caja negra para los otros.

**Decide\_action.pl**

El módulo decide\_action.pl es el módulo en el cual se implementará la estrategia en la siguiente etapa. Por el momento, y de manera preliminar, se decidió incorporar la estrategia de movilidad dada por el agent\_barb.pl pero con sutilezas de cambio que lo hacen más competitivo.

La esencia del agente se basa en decidir acciones las cuales son los *átomos* de la estrategia a seguir. Para nuestro agente la estrategia seguida es la siguiente, estableciendo en los ítems la prioridad de las acciones según se presentan y siendo los cuerpos del predicado decide\_action/1:

* En el caso de que exista un agente en el rango de ataque, se toma la acción de atacar a ese agente de manera reactiva y sin realizar ningún tipo de análisis de la información almacenada hasta el momento. En la entrega siguiente se utilizará la información almacenada para determinar mejor la estrategia de ataque a otro agente. Por otro lado, para determinar si un agente está en una situación vulnerable de ataque, se recorren las 4 posibles celdas de ataque de la percepción, y si alguna de estas contiene un agente, se decide atacarlo.

Imagen del agente atacando a otro

* Si en la celda en la que se encuentra el agente, también hay una posada, entonces el agente decide quedarse hasta llenar su *stamina* al máximo antes de salir. En el caso de que aparece un agente en la posición atacable, entonces decide atacarlo mientras recarga su energía.

Imagen de agente sobre una posada

Las siguientes versiones del predicado decide\_action/1 corresponden a decidir girar a la izquierda o derecha (la derecha o izquierda se determina mediante el punto cardinal al cual está mirando actualmente) dependiendo si en esas direcciones hay un tesoro o una posada:

* Al fijarse en la percepción, si encuentra que entre los atributos de la celda que se encuentra a su derecha hay una posada o un tesoro, entonces toma como decisión girar a la derecha.
* Análogamente al anterior, si entre los atributos de la celda de la izquierda, encuentra q existe un tesoro o una posada, decide girar a la izquierda como siguiente acción a realizar.
* En el caso de que en la posición donde se encuentra el agente, también exista un tesoro, el agente realizará *tomar* como acción. Esto hará que el agente tome y guarde en su inventario este tesoro.
* Si ninguna de las acciones anteriores no puede realizarse, se toma como acción por defecto realizar un paso a la posición de enfrente a la cual el agente se encuentra mirando en ese momento.

**auxiliar\_predicates.pl**

Este módulo se encarga de realizar la muestra por pantalla del estado interno del agente. En las impresiones se muestran los siguientes atributos:

* Nombre del agente;
* Meta actual del agente, respecto a la estrategia a seguir en los próximos turnos;
* Última acción realizada;
* Muestra la lista de agentes conocidos, con el historial de acciones realizadas mientras estuvo en el rango de vista;
* Muestra la lista de posadas que recuerda con sus posiciones;
* Muestra la lista de tesoros que vio y todavía faltan ser juntados;

Cada uno de los datos almacenados que posee una estructura particular, la cual se recorre debidamente para determinar los datos almacenados y luego imprimirlos por pantalla. Esta impresión se realiza adecuadamente para respetar el formato virtual, es decir, imprimir el correspondiente dato virtual con su semejante carácter o caracteres.

Por otro lado, también se incorporó en este módulo un predicado que recorre la percepción recibida por el entorno y la muestra por pantalla. Esto sirve para comparar los datos almacenados, con los datos que maneja el entorno y chequear que la actualización de la memoria se hace correctamente.

**extras\_for\_agents.pl**

En el módulo extras\_for\_agents.pl, se incorporó el código dado por la cátedra, el cual determina para ciertas posiciones datos de utilidad, ya sea, las posiciones cardinales, posiciones al girar 90º, etc.

**agent\_emig.pl**

En este sector del agente, se agrupan los predicados que realizan la puesta en marcha del agente en el entorno. En primer lugar, se cargan los módulos adicionales del agente explicados en el gráfico y luego de que está todo correcto, comienza a ejecutarse el ciclo del predicado *run.*

En este ciclo se realizan las siguientes acciones:

* acciones de solicitud de percepción;
* muestra de la percepción actual recibida;
* se actualiza el estado interno del agente con la precepción tomada del entorno dada por el rango de visión del agente;
* luego de actualizado el estado, se decide realizar un volcado por pantalla de la memoria actual del agente para reflejar que los datos tomados fueron realmente mapeados en su estado interno;
* terminado esto y con el estado interno actualizado, se decide tomar la decisión de la acción a realizar en el siguiente turno. Esto hace que dependiendo de los datos adquiridos, modifique potencialmente el objetivo que el agente tendrá y su acciones a realizar;
* por último, se efectiviza la acción mediante el predicado *do\_action/1* y se realiza la invocación nuevamente al predicado *run* para realizar un nuevo ciclo de ejecución del agente en el entorno.

Otros predicados que se pueden encontrar aquí son start\_ag/0 y start\_ag\_instance/1. Estos predicados incorporan al funcionamiento del agente porque hacen que este se conecte al entorno de combate dado por la cátedra. El segundo de los predicados, permite incorporar varias instancias de un mismo agente, dándole un identificador distintivo a cada uno. Por otro lado, el primer predicado solo deja agregar un solo agente de esta clase.