Proyecto de Prolog: Hive

Victor Manuel Cardentey Fundora C411
David Guaty Domínguez C412

Estructura del proyecto

El proyecto consta de 4 módulos:

1. app.pl: El módulo encargado de la interacción con el usuario mediante una interfaz gráfica realizada en pce.

2. board.pl: El módulo encargado de modelar la estructura actual de las piezas colocadas. Brinda información importante sobre el estado actual del juego.

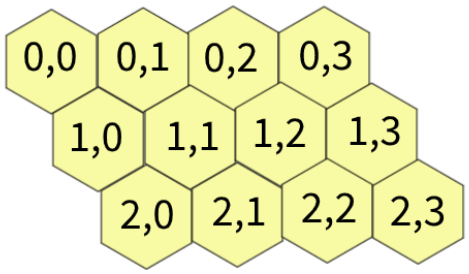
3. bugs.pl: EL módulo encargado del manejo de los distintos tipos de insectos. Puede calcular los posibles movimientos de cada insecto.

4. cpu.pl: El módulo encargado de implementar el jugador no humano.

board.pl

Modelado del tablero

Aunque el juego no presenta un tablero, el modelado de las posiciones de las piezas sitúa las piezas en un grid hexagonal infinito. Para modelar un grid hexagonal se tienen varias opciones, en este proyecto se eligió rotar el eje x del grid de tal forma que cada fila de hexágonos está desplaza a la derecha por medio hexágono con respecto a la anterior. Por ejemplo, un grid hexagonal de 4x4 se vería así:



El módulo board

Teniendo en cuenta la definición de grid hexagonal anterior, se implementó el módulo board para obtener información acerca del estado actual del juego.

Para saber si dos casillas son adyacentes se tiene el predicado adyacent(X1, Y1, X2, Y2), que triunfa si el hexágono con coordenadas X2,Y2 es adyacente al hexágono X1, Y1.

Para llevar el estado del juego, board.pl tiene varios predicados dinámicos que van cambiando de acuerdo a si se coloca una pieza nueva o si se mueve una pieza.

Para saber si en la celda con coordenadas X,Y existe un insecto en juego se tiene el predicado bug/5.

bug(C,T,X,Y, S) triunfa si existe un bug de color C(con color se hace referencia a un tipo de jugador), de tipo T(hormiga, reina,etc), que está en la posición X,Y. La variable S se refiere a la posición del insecto dentro de la pila de insectos que se puede formar, por ejemplo un escarabajo que esté encima de una reina tendrá valor S =1, mientras que la reina tendrá valor S=0.

A continuación se explica como se colocan piezas.

Celdas fronteras

Para colocar un insecto en juego, este tiene que ser adyacente a algún insecto de la colmena, si no se cumple esto se rompería la colmena. Por lo que se tiene la definición de **celda frontera**. Una celda vacía con coordenadas (X,Y) es **celda frontera** si es adyacente a una celda no vacía.

Para saber si una casilla está vacía se tiene: *empty(X,Y):- \+ bug(_,_, X, Y, _)* .

Manteniendo almacenadas las celdas fronteras y colocando insectos

Para evitar calcular todas las celdas fronteras cada vez que se vaya a colocar un insecto, se tiene el predicado dinámico frontier/2, frontier(X,Y) triunfa si la celda con posicion (X,Y) es frontera.

Para colocar un insecto en juego se tiene el predicado placeBug(C,T,X,Y): coloca el insecto con color C, de tipo T, en la posición X,Y.

Al colocar un nuevo insecto en la celda X,Y que es frontera, se expande la frontera de la colmena, almacenando así las nuevas casillas fronteras. En el placebug se tiene la siguiente línea: *forall(emptyAdyacent(X,Y,X1,Y1), setFrontier(X1,Y1))*. Tambien se setea un nuevo bug, o sea se añade el predicado bug(C,T,X,Y,0) en la base de datos.

Nota: El placeBug hace un poco más, porque también sirve para el movimiento de las piezas. El placeBug del módulo escoge la altura de la celda con el predicado getCellHeight(X,Y,H) donde se obtiene en H cuantas piezas hay apiladas en la posición X,Y. El insecto con mayor altura dentro de una celda con altura H es H-1. Luego coloca un bug en la altura H seteando bug(C,T,X,Y,H).