GUIÓN 2
Exploración y Búsquedas

Inteligencia Artificial

Manuel Ruiz Fernández **26518655T**

mrf00020@red.ujaen.es

Juan Bautista Muñoz Ruiz **26516720C**

jbmr0001@red.ujaen.es

M21B13a

Descripción de la estrategia:

La estrategia usada necesita de dos hashmaps con un pair de integer cómo clave y un grid cómo elemento. Uno de los mapas es el de las celdas exploradas y otro el de las celdas que temporalmente hemos visitado. Además, tendremos una pila con los movimientos que hemos hecho.

Para obtener el movimiento, en primer lugar añadiremos la celda a ambos mapas en caso de que no esté. Si no estaba en el mapa de explorados además de añadir la celda, incrementaremos el contador de celdas exploradas.

Tras esto, aunque no se trate de una estrategia de bomba. Creamos un random con el que sacaremos un número de 1 a 50, si sale un 5 se pone una bomba.

Una vez hecho esto, pasamos a ver hacia dónde nos moveremos y por qué.

En primer lugar se comprueban las celdas cercanas, si se puede ir a ellas y si estas han sido visitadas con el map de las exploradas, añadiendo a una lista el movimiento que nos llevaría a la casilla no visitada. En caso de que más de un movimiento haya sido añadido a la lista usamos un entero random entre el número total de movimientos y sacamos el movimiento de esa posición. Antes de devolver ese movimiento, se añade a la pila de movimientos.

En caso de que las celdas cercanas hayan sido visitadas anteriormente pasamos a comprobar de la misma forma el map de las celdas que temporalmente hemos visitado. En caso de que también hayan sido visitadas todas por el map temporal pasamos a sacar el último movimiento de la pila de movimientos para dar marcha atrás.

Si finalmente hasta la pila de movimientos está vacía renovamos el mapa de celdas temporalmente visitadas y con esto seguro que al menos 1 de los movimientos será posible y saldremos de ese atasco.

En caso de morir renovaremos tanto la pila de movimientos cómo el mapa de celdas por si aparecemos en un lugar poco conveniente (lugar con todo ya explorado).

Ejecuciones:

Ejecución 20x20 60 segs	Quesos	Pasos	Celdas Exploradas	Ratio de Exploración
1	1	221	173	0,432
2	0	221	178	0,445
3	3	223	154	0,385
4	0	219	140	0,35
5	0	219	142	0,355

Ejecución 20x20 120 segs	Quesos	Pasos	Celdas Exploradas	Ratio de Exploración
1	0	437	281	0,702
2	3	442	283	0,707
3	0	435	259	0,647
4	1	436	310	0,775
5	1	435	287	0,717

Ejecución 40x20 120 segs	Quesos	Pasos	Celdas Exploradas	Ratio de Exploración
1	0	436	313	0,391
2	0	436	243	0,30375
3	0	436	320	0,4
4	1	437	317	0,396
5	1	438	340	0,425

Ejecución 40x20 240 segs	Quesos	Pasos	Celdas Exploradas	Ratio de Exploración
1	1	874	549	0,686
2	1	869	582	0,727
3	2	866	548	0,685
4	0	919	378	0,472
5	2	867	588	0,735

Análisis de los resultados:

Aunque el ratio de exploración es en ocasiones alto el número de quesos obtenidos deja mucho que desear siendo muy bajo con 0 quesos recogidos en numerosas ocasiones.

M21B13DFS

Descripción de la estrategia:

En esta estrategia se añade estructuras de datos y variables usadas en la exploración una *LinkedList<int>* para guardar los movimientos tras la búsqueda DFS, un *HashMap<Grid>* de celdas visitadas por el DFS, un boolean que indica si el queso ha sido encontrado y un boolean que indica si ha nacido otro queso a las

Move(): Dentro de la función, si la casilla actual no está explorada la metemos en el HashMap de celdas y de celdas temporales al igual que en la exploración.

Seguidamente encontramos la siguiente toma de decisiones:
-Si la casilla en la que se encuentra el queso ha sido explorada y se ha respawneado un nuevo queso llamamos al DFS() y sacamos el último movimiento de la lista.
-Si la casilla en la que se encuentra el queso ha sido explorada y pero el queso sigue siendo el mismo devolvemos un movimiento de la lista.
Si por el contrario la celda en la que se encuentra el queso no está explorada devolvemos una llamada a la función explorar(). Función que contiene toda la exploración desarrollada y explicada en el M21B12B1.

DFS(): Se trata de un algoritmo DFS recursivo que calcula los movimientos hacia el objetivo tras el recorrido primero en profundidad usando las celdas recorridas en la exploración. Por lo que para su correcto funcionamiento se necesita explorar el mapa en gran medida.

Dentro del DFS se crea una lista de movimientos y de celdas donde se almacenarán las celdas y movimientos adyacentes. Cada llamada a esta función se hace un push de la celda actual, un push del movimiento que se le ha pasado al DFS y una inserción en el HashMap de celdas visitadas por el DFS de la celda actual.

Si la celda actual se corresponde con la del queso establecemos la variable *encontrado* como verdadera. En caso contrario calculamos las celdas, movimientos adyacentes y los almacenamos en sus respectivas listas. Una celda o movimiento será adyacente si se encuentra en las celdas visitadas de la exploración y no ha sido visitada por el DFS.

Por último para cada elemento de la lista de adyacentes si no ha sido visitado por el DFS se hace una llamada al propio DFS con la celda adyacente, el movimiento para llegar a esta y el queso como parámetros.

En caso de que un nodo no tenga más hijos adyacentes se hará un pop() de la lista de

movimientos. De esta manera se borran de la lista los movimientos que no conducen al queso.

Ejecuciones:

Laberinto	Ejecución	Tipo	Quesos	Pasos	Celdas exploradas	Ratio de exploración	Ratio de repetición
20x20 300 segs	1	DFS	8	1103	322	0,805	3,425
	2	DFS	6	1090	364	0,91	2,994
	3	DFS	5	1084	379	0,9475	2,860

Análisis de los resultados:

Se ve una clara mejoría con respecto a la exploración, tanto en el notable número de quesos como en el muy alto ratio de exploración. Aunque es cierto que para ir hacia los quesos necesita explorar el mapa en gran medida por lo que cuanto más grande sea el mapa más trabajo le costará.

M21B13B1

La primera estrategia de lanzamiento de bombas se basa en la existencia de tres o cuatro caminos disponibles, es decir, un cruce. Ya que en las celdas en las que se cruzan varios caminos hay más probabilidad de que pase otro ratón.

Por lo que en la función *move()* solo se soltará una bomba en caso de que el ratón pueda moverse en tres o cuatro direcciones posibles.

Para asegurarnos de que no se queda solo poniendo bombas pondremos un contador de pasos que cada vez que se ponga una bomba volverá a 10 e irá disminuyendo con cada paso.

M21B13B2

La segunda estrategia de lanzamiento de bombas se basa en usar la necesidad de los jugadores de acercarse a un queso y cogerlo. Al hacer esto mismo nuestro DFS, cuando estemos a 1 casilla del queso colocaremos la bomba y cogeremos el queso. Aunque el resto de ratones no lleguen a coger ese queso, estarán en las inmediaciones y es bastante probable que pisen la bomba.

Para ello en el método move se comprobará la distancia con el queso y en caso de estar en el mismo X o Y y en +-1 de distancia en el otro eje se soltará la bomba.

Ejecuciones uno contra uno:

Laberinto	Ejecución	Tipo	Quesos	Pasos	Celdas exploradas	Ratio de exploración	Ratio de repetición
20x20 300 segs	1	B1	5	1128	331	0,8275	3,407
		B2	8	1143	298	0,745	3,835
	2	B1	11	1155	330	0,825	3,5
		B2	7	1213	252	0,63	4,813
	3	B1	4	1192	379	0,947	3,145
		B2	2	1178	376	0,94	3,132

Ejecuciones por separado con 5 exploradores:

Laberinto	Ejecución	Tipo	Quesos	Pasos	Celdas exploradas	Ratio de exploración	Ratio de repetición
20x20 300 segs	1	B1	5	1182	339	0,847	3,48
		B2	7	1147	344	0,860	3,33
	2	B1	5	1159	347	0,867	3,34
		B2	5	1227	317	0,792	3,87
	3	B1	7	1222	349	0,872	3,50
		B2	6	1211	333	0,832	3,63

Laberinto	Ejecución	Tipo	Quesos	Pasos	Celdas exploradas	Ratio de exploración	Ratio de repetición
40x20 300 segs	4	B1	2	1226	614	0,767	1,996
		B2	4	1073	430	0,537	2,495
	5	B1	2	1134	482	0,602	2,352
		B2	4	1143	426	0,533	2,683
	6	B1	3	1120	440	0,55	2,545
		B2	3	1125	449	0,561	2,505

Análisis de los resultados:

En la ejecución por separado se obtienen resultados parecidos por lo que no podemos inferir cual de los dos es más óptimo poniendo bombas. Sin embargo en la ejecución uno contra otro el resultado también varía, pero podemos observar que el B1 parece funcionar un poco mejor.

Aunque al quedarse sin bombas a apenas un minuto de empezar no podemos valorar mucho si una estrategia determina ganar o no