PROYECTO DE LA ASIGNATURA DE INTELIGENCIA ARTIFICIAL Y SISTEMAS INTELIGENTES 2022/2023

Alumno:
-Sergio Orgaz Bravo

1_.Resumen del problema de la práctica:

-La práctica tiene como principal objetivo desarrollar una inteligencia artificial, en la que sus algoritmos sean capaces de resolver una serie de problemas propuestos de la manera más eficiente y en el menor tiempo posible.

Siendo más específico en la parte de los problemas propuestos, se refieren a un robot el cual se encuentra en un laberinto y para salir de él debe recoger unas monedas hasta llegar a un valor previamente indicado para finalmente acabar buscando la salida del laberinto.

2_.Descripción de la instalación, interfaz y manejo de la práctica:

El proyecto ha sido desarrollado en “Eclipse” en la interfaz de C++, esta es la página oficial para poder descargar el programa https://www.eclipse.org/downloads/.
En su interfaz podemos ver las distintas clases del proyecto mediante una barra desplegable.

Dentro de cada clase podemos apreciar dos tipos de archivos, .h y .cpp, el .h contiene todas las cabeceras de los métodos de la propia clase y todas las herencias de otras clases o librerías. En el .cpp encontramos la definición de todos los métodos de la clase, tanto constructores como destructores.

Para el desarrollo del proyecto no ha sido necesaria la implementación de ninguna librería adicional.

Para ejecutar el proyecto debemos ejecutar su main(), el cual se encuentra en el archivo IASI.cpp, en el cual nos aparecerá un menú en el cual tendremos que elegir el algoritmo que resolverá nuestro proyecto.

3_.Descripción de los algoritmos de resolución usados en la solución

Los algoritmos empleados han sido:

*Escalada simple:

Desde el nodo padre se prueba un nuevo operador y se evalúa su coste, si el coste del nuevo operador es mejor que el del padre este pasará a ser el nuevo nodo padre. en el caso de que ningún nodo sea mejor que el padre se para el algoritmo

*Escalada simple estocástico

Desde el nodo padre se prueba un nuevo operador, se evalúa su coste y si mejora un valor determinado , si el coste del nuevo operador es mejor que el del padre este pasará a ser el nuevo nodo padre. en el caso de que ningún nodo sea mejor que el padre se para el algoritmo

*Máxima pendiente

Desde el nodo padre se prueban todos los nodos posibles a los que se puede acceder y se evalúan, una vez generados se evalúan y se selecciona el mejor nodo y pasa a ser el nodo padre. Si ningún nodo generado es mejor que el padre se para el algoritmo

4_.Descripción de las soluciones seguidas para resolver el problema:

Para resolver el problema se han planteado tres algoritmos de búsqueda con heurística, los cuales han sido Escalada Simple, Escalada Simple Estocastico y Máxima Pendiente.
He implementado dos heurísticas, la primera de ellas ha sido una la cual busca la distancia de un nodo inicio a un nodo destino (una moneda o una salida).

La segunda heurística ha sido una exclusiva para buscar monedas, la cual busca en el vector la moneda de mayor valor cercana (destino) a la posición del robot, una vez alcanzado el valor de las monedas objetivo se busca la salida con la heurística de menor distancia.

-Escalada Simple:

En este algoritmo lo primero que creamos es un nuevo tablero y seleccionamos el laberinto que queremos resolver una vez seleccionado este laberinto buscamos la coordenada del robot y la coordenada de la salida. Si no se ha encontrado una salida en el laberinto el programa nos indica que no ha encontrado salida, el tiempo que ha tardado en ejecutarse el proyecto y nos indica que ha generado 0 nodos.

En el caso de que si exista una salida, el algoritmo busca la posición de la moneda más cercana a la posición del robot, independientemente del valor, una vez encontrada la

moneda va generando movimientos, así sucesivamente hasta llegar al valor de las monedas objetivos, estos movimientos se generan siguiendo la explicación del algoritmo de escalada simple se generan nodos hasta encontrar uno mejor que el padre, si no se encuentra ninguno mejor se para el algoritmo indicando los nodos generados, el tiempo de ejecución y los movimientos realizados hasta el momento. Una vez alcanzado el valor de la monedas objetivos se avanza hasta la salida repitiendo el mismo proceso

-Escalada simple Estocástico:

Este algoritmo realiza lo mismo que el anterior añadiendo la condición de que el nodo hijo mejore al padre en un cierto valor que indicamos al algoritmo por pantalla.

-Máxima Pendiente:

En este algoritmo lo primero que creamos es un nuevo tablero y seleccionamos el laberinto que queremos resolver una vez seleccionado este laberinto buscamos la coordenada del robot y la coordenada de la salida. Si no se ha encontrado una salida en el laberinto el programa nos indica que no ha encontrado salida, el tiempo que ha tardado en ejecutarse el proyecto y nos indica que ha generado 0 nodos.

En el caso de que si exista una salida, el algoritmo busca la moneda de mayor valor más cercana al robot y va generando todos los posibles nodos hijos, una vez generados avanzamos hasta aquel hijo que se encuentre más cerca de la moneda objetivo , si hemos alcanzado el valor de las monedas objetivo avanzamos hacia la salida. En el caso de que ninguno de los nodos generados sea mejor que el nodo padre se para el algoritmo.

5_.Resultados obtenidos en las baterías de pruebas para cada una de las soluciones:

LABECOIN 1:

Algoritmo	Movimientos	Tiempo	Nodos
Escalada Simple	I-I-AR-AR (sin sol)	2.1^-5s	18
ESEstocastico	I-I (sin sol)	7.1^-5s	16
Maxima Pendiente	ARD-AR-D-D-D-D (sin sol)	3.1^-5 s	15

LABECOIN 2:

Algoritmo	Movimientos	Tiempo	Nodos
Escalada Simple	D-AR-AR-AR-I-I-I-I-I (sol)	4.7^-5	34
ESEstocastico	D-ARI-ARI-ARI-I-I (sol)	0.000156	37
Maxima Pendiente	D-ARI-ARI-I-I (sol)	8.5^-5s	39

LABECOIN 3:

Algoritmo	Movimientos	Tiempo	Nodos
Escalada Simple	AB (sin sol)	4.7^-5s	10
ESEstocastico	AB (sin sol)	0.000133	10
Maxima Pendiente	AB-ABI-D-D-D-AB-ABI-ABI-ABI-D-D-D-D-D-D-D-D-ARI-ARD-AR-AR-AR-AR-AR-ARD (sol)	0.000179 s	104

LABECOIN 4:

Algoritmo	Movimientos	Tiempo	Nodos
Escalada Simple	D-D-D-D (sin sol)	4.2^-5s	20
ESEstocastico	D-D-D-D(sin sol)	0.000135s	20
Maxima Pendiente	D-D-D-D-AR-AR-I-AB-I-AB-I-AB-ABD-D-D-D (sin sol)	7.6^-5 s	69

LABECOIN 5:

Algoritmo	Movimientos	Tiempo	Nodos
Escalada Simple	AB	7.1^-5	10
ESEstocastico	sin sol	0.000136s	8
Maxima Pendiente	ARI-ARI-D-D-D-AB-AB-I-AR (sin sol)	5.2^-5s	34

LABECOIN 6:

Algoritmo	Movimientos	Tiempo	No dos
Escalada Simple	AB	7.1^-5s	10
ESEstocastico	sin sol	0.000208	8
Maxima Pendiente	ARI-ARI-D-D-D-AB-AB-I-AR (sin sol)	7.9^-5 s	34

LABECOIN 7:

Algoritmo	Movimientos	Tiempo	Nodos
Escalada Simple	AR-I-AB-AB-AB-D-D-D-AR-AR-I-I (sin sol)	0.00017	38

ESEstocastico	ARI(sin sol)	7.5^-5s	14
Maxima Pendiente	ARI-ARD-ARD-ARD-D-ABI-ABI-ABI-I-AR-ARI-ABI-I (sin sol)	5.5^-5s	74

LABECOIN 8:

Algoritmo	Movimientos	Tiempo	Nodos
Escalada Simple	I-I-I (sin sol)	2^-5s	20
ESEstocastico	I-I-I(sin sol)	0.000111	14
Maxima Pendiente	I-I-I-AR-AR-AR (sin sol)	8.7^-5s	12

LABECOIN 9:

-Algoritmo	Movimientos	Tiempo	Nodos
Escalada Simple	Sin movimientos	3^-6s	0
ESEstocastico	Sin movimientos	0.000132	0
Maxima Pendiente	Sin movimientos	3^-6s	0

LABECOIN 10:

Algoritmo	Movimientos	Tiempo	Nodos
Escalada Simple	Sin movimientos	3`-6 s	0
ESEstocastico	Sin movimientos	0.000121	0
Maxima Pendiente	Sin movimientos	3^-6 s	0

6 _. Conclusiones sobre los resultados para cada una de las técnicas empleadas para la resolución empleadas.

De las técnicas de resolución saco la conclusión de que no son las mejores para alcanzar una solución final, pero sí considero que son las más rápidas.

Los algoritmos de escalada buscan únicamente una posibilidad siempre que esta sea mejor que el nodo padre, y esto en ocasiones nos puede hacer llegar a un callejón sin salida provocando que nuestro algoritmo se detenga sin encontrar una solución a pesar de que si exista una.

Por esta misma razón considero que es mucho mejor la implementación de otro tipo de algoritmo como “Primero Mejor” o “Algoritmo A*”.