Caso grupal: Resolución del problema del puzzle-8 mediante búsqueda heurística

**Objetivos** de la actividad

Con esta actividad vas a conseguir implementar la estrategia de búsqueda heurística A\* para la resolución del problema del puzzle-8.

**Descripción** de la actividad

El puzzle-8 es un juego de un único jugador que consiste en un tablero de 9 posiciones (3x3), y ocho fichas numeradas del 1 al 8, situadas en el tablero, quedando una posición del tablero vacía. El objetivo del juego es alcanzar una determinada disposición de las fichas a partir de una disposición inicial realizando sólo los movimientos permitidos, que son mover una ficha adyacente a la posición vacía de forma horizontal o vertical, ocupando la posición vacía y quedando en su lugar vacía la casilla ocupada anteriormente por la ficha movida.

En esta actividad has de utilizar la estrategia de búsqueda heurística A\* con el fin de resolver el problema del puzzle-8. Utiliza como heurística el número de fichas mal colocadas respecto al estado objetivo. Considera que el coste de cada movimiento es 1.

El estado inicial del puzzle es el siguiente:

|  |  |  |
| --- | --- | --- |
| 2 | 8 | 3 |
| 1 | 6 | 4 |
| 7 |  | 5 |

El estado objetivo es el siguiente:

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| 8 |  | 4 |
| 7 | 6 | 5 |

Debes desarrollar en el lenguaje de programación Python una implementación del algoritmo A\* que desarrolle el árbol de búsqueda desde el estado inicial al estado objetivo, anotando cada nodo con el valor de la función de evaluación del nodo *f(n)*. Además, en cada iteración del algoritmo debes indicar claramente el nodo que ha sido expandido, el contenido de la lista abierta y de la lista cerrada de nodos (estados) con su valor de función de evaluación.

Se deberán crear tantas clases o estructuras de datos como sean necesarias para representar el espacio de estados y los nodos de exploración del árbol.

El código deberá ejecutarse una vez y alcanzar el resultado, indicando la secuencia de acciones a realizar para alcanzar el objetivo utilizando una notación sencilla. Por ejemplo: «mover 7 hacia la derecha» o «mover 5 hacia la izquierda».

El programa en Python desarrollado debe ser un trabajo original del estudiante.

**Rúbrica**

|  |  |  |  |
| --- | --- | --- | --- |
| Actividad Grupal: Resolución del problema del puzzle-8 mediante búsqueda heurística (valor real: 3 puntos) | Descripción | Puntuación máxima  (puntos) | Peso  % |
| Criterio 1 | El código implementa correctamente la lógica del algoritmo A\* (sigue el orden correcto en el desarrollo del árbol de búsqueda) | 4 | 40% |
| Criterio 2 | El cálculo de la función de evaluación de los nodos f(n) es correcta | 2 | 20% |
| Criterio 3 | El contenido de la lista abierta y de la lista cerrada para cada iteración es correcta | 2 | 20% |
| Criterio 3 | El resultado obtenido, es decir la secuencia de acciones a realizar para alcanzar el objetivo, es correcto | 2 | 20% |
|  |  | **10** | **100 %** |

**Extensión** máxima de la actividad: 10 páginas.

**Organización y gestión de equipos**

En el foro «Pregúntale al profesor» de la asignatura encontrarás un nuevo tema específico para la organización de equipos donde el profesor explicará todos los detalles.

Una vez cerrado el equipo de trabajo os podéis poner en contacto a través de vuestras cuentas @comunidadunir.net y comenzar a trabajar. Puedes ampliar la información sobre el trabajo en equipo, consultando los [**Tutoriales de trabajo en grupo**](http://tv.unir.net/secciones/3967/4883/0/0/0/0/).

**IMPORTANTE:** Aquellos **estudiantes que no comiencen su trabajo dentro de los 7 primeros días**, contados a partir del día de inicio de la actividad, **quedarán** **excluidos** de la actividad, no pudiendo tomar parte en ella. Se trata de una actividad colaborativa, por lo que unos estudiantes no pueden beneficiarse del trabajo que hayan realizado sus compañeros.

**Entrega de la actividad grupal**

Al finalizar la actividad grupal, todos los miembros del equipo entregarán la misma actividad a través del apartado «Envío de actividades» del aula virtual. El documento a entregar, debe ir nombrado así:

APELLIDO1\_APELLIDO2\_NOMBRE\_Titulo\_actividad (sin tildes ni apóstrofes ni ningún otro carácter que pudiera resultar conflictivo).

**Todos los miembros del equipo deben hacer la entrega en el aula virtual y deben adjuntar el mismo documento.**

Indica en la actividad el nombre de todos los componentes del equipo y cumplimenta la siguiente tabla de valoración individual:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Sí | No | A veces |
| Todos los miembros se han integrado al trabajo del grupo |  |  |  |
| Todos los miembros participan activamente |  |  |  |
| Todos los miembros respetan otras ideas aportadas |  |  |  |
| Todos los miembros participan en la elaboración del informe |  |  |  |
| Me he preocupado por realizar un trabajo cooperativo con mis compañeros |  |  |  |
| Señala si consideras que algún aspecto del trabajo en grupo no ha sido adecuado |  |  |  |

### Definición de la clase Nodo

Para el desarrollo del ejercicio se realizará una implementación mediante es uso de clases tipo nodo. Para esto se crea la clase que tiene la siguiente información:

\* El atributo \*\*nodo\*\* el cual tiene representa el id del nodo.

\* El atributo \*\*puzzle\*\* es una matriz nxn que contiene los números que se representan en el juego.

\* El atributo \*\*accion\*\* representa el tipo de movimiento el cual puede tomar los valores: arriba,abajo,izquierda,derecha

\* El atributo \*\*ficha\*\* identifica la ficha que fue movida.

\* El atributo \*\*nodoPadre\*\* determina el padre de un nodo y este sirve para determinar la ruta hacia el nodo meta.

\* El atributo \*\*nodoshijos\*\* son los nodos hijos que se generan en función del nodo padre.

Métodos:

\* El método \*\*ordenar\_lista\_heuristica\*\* permite ordenar los nodos hijos por el número de fichas mal colocadas respecto al estado objetivo.

\* El método \*\*calcular\_heuristica\*\* determina el número de fichas mal colocadas respecto al estado objetivo.

### Definición de la clase PuzzleClass

Se crea una clase que representa el juego como tal, esta clase implementa métodos auxiliares para la creación y manejo del puzzle.

Métodos:

\* El método \*\*buscar\_espacio\_vacio\*\* permite identificar los índices fila y columna del espacio vacío.

Nota: El espacio está representado por el número: 0.

\* El método \*\*pos\_adyacente\*\* permite identificar todas las posiciones adyacentes al espacio vacío en el tablero.

\* El método \*\*crear\_puzzle\_hijos\*\* permite crear todas las combinaciones nodos hijos con sus respectivos movimientos por cada ficha una vez determinada la posición vacía.

\* El método \*\*crear\_nodos\_hijos\*\* asigna los nodos hijos para cada nodo principal.

\* Los métodos \*\*imprimir\_resultado\_nodo\*\* y \*\*imprimir\_meta\*\* permiten imprimir los resultados de la ejecución del algoritmo A\*.

\* El método \*\*resolver\_puzzle\*\* representa la búsqueda A\* basada en heurística, este algoritmo utiliza este paradigma y por tanto debe crear los nodos hijos y de igual manera la llamada a la ejecución de la función heurística sobre estos nodos.

\* Se crean las constantes que serán utilizadas para el desarrollo del ejercicio.

\* La variable \*\*maxima\_profundiad\*\* establece un valor máximo de profundidad para evitar bucles infinitos.

\* La variable \*\*imprimir\*\* la cual permite imprimir los resultados durante la ejecución del algoritmo A\*.

\* La variable \*\*espacio\_vacio\*\* identifica el valor vacío en el puzzle.

### Definición métodos para graficar nodos

Métodos:

\* El método \*\*crear\_ruta\_nodos\*\* permite crear la ruta desde los nodos inicial y final, para esto realizar un ordenamiento navegando por el nodo padre.

\* El método \*\*graficar\_ruta\_nodos\*\* permite visualizar el resultado de los nodos y sus acciones cada flecha corresponde a una acción para llegar al objetivo.

Estimado Cesar,

Le deseo un buen día en el desarrollo de sus actividades.

Primero, agradecer por su feedback detallado respecto de la actividad grupal. En el cual se menciona: “En el algoritmo A\*, la primera comprobación de nodos visitados sobra. Antes de insertar en nodos abiertos se debe verificar que el nodo no está previamente.”

En el trabajo en grupo realizamos una implementación análoga a la descrita en la documentación de UNIR en la cual se ejemplifica el siguiente fragmento de código:

Para cada hijohijo en Grafo .hijos nodo hacer

\*\*si hijo no esta marcado como visitado\*\*

marcar hijo como visitado

hijo .padre = nodo

colaAbierta .add hijo

fin si

Adicionalmente, esta lista de nodos abiertos como se muestrea en la presentación vistas en clase “Ejemplo Búsqueda heurística con Algoritmo A\*”, pagina 6 al 11; como se puede apreciar en la imagen 1, un nodo puede estar repetido varias veces dentro de los nodos abiertos por ejemplo Barcelona y Bilbao. En la actividad, para evitar el acceder a nodos abiertos se hace uso de la lista de visitados que incluimos en el código.

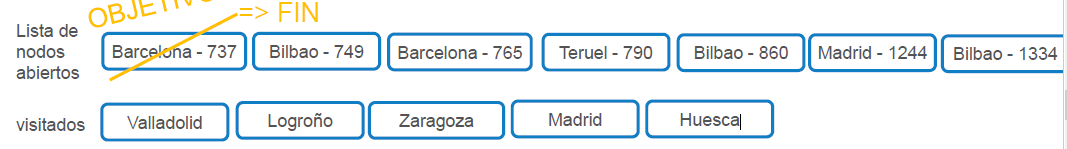


Imagen 1

Finalmente, solicitarle por favor nos aclare la duda ya que realizamos varias pruebas (distintas configuraciones) contra otros ejemplos publicados en internet y los resultados fueron correctos.

Bibliografía: