UNIVERSIDADE DA CORUÑA

Grado en Ingeniería Informática
Sistemas inteligentes

Práctica 1: Estrategias de búsqueda
Curso 2022/2023

En esta práctica desarrollarás en Java varias estrategias de búsqueda y las utilizarás para resolver un problema real. Para facilitar el desarrollo de la práctica, se proporciona un proyecto de IntelliJ IDEA con código de ejemplo que incluye una primera implementación: la de la Estrategia Básica 4 vista en teoría.

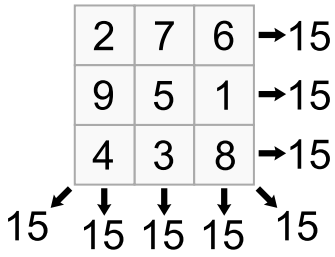
- **Ejercicio 1: Haz las siguientes modificaciones sobre el código proporcionado.**

  a) Crea una clase `Nodo` según lo descrito en teoría (D46). Adapta `Estrategia4` para que registre los estados que va explorando utilizando `Nodos`.
  Cambia el método `soluciona` de la clase `EstrategiaBusqueda` para que devuelva un array de `Nodos(Nodo[])`. Modifica la implementación de `soluciona` en `Estrategia4` para que, cuando encuentre una solución, devuelva la lista de nodos que representan los estados recorridos desde el estado inicial hasta la meta encontrada. Implementa para ello en `Estrategia4` un método `reconstruye sol` como el descrito en teoría.

  b) `Estrategia4` falla cuando llega a un estado sin sucesores. Sin embargo, es posible que estados por los que había transitado previamente todavía tengan sucesores sin explorar. La estrategia `Busqueda Grafo` vista en teoría soluciona esto añadiendo una frontera en la que almacena los estados sucesores encontrados mientras no son explorados. Implementa una clase `EstrategiaBusquedaGrafo` (que herede de `EstrategiaBusqueda`) que implemente dicha estrategia (te recomendamos utilizar `Estrategia4` como plantilla).

- **Ejercicio 2: Buscamos solucionar el siguiente problema utilizando estrategias de búsqueda:**

  Un cuadrado mágico de NxN es una matriz que contiene los números entre 1 y N^2 dispuestos de tal manera que la suma de los elementos de cada una de sus filas (o de sus columnas o de sus diagonales principales) es siempre la misma: $\frac{N(N^2+1)}{2}$



Ejemplo de cuadrado mágico 3x3

Queremos desarrollar un agente capaz de, a partir de un cuadrícula parcialmente rellena, completar el cuadrado mágico (si es que es posible), respetando los valores suministrados originalmente.

Ejemplos:

Estado inicial:

4	**9**	**2**
3	**5**	
	1	

Estado meta:

4	**9**	**2**
3	**5**	**7**
8	**1**	**6**

2		

2	**9**	**4**
7	**5**	**3**
6	**1**	**8**

u otro cuadrado mágico de 3x3

2			
	1		

2	**8**	**15**	**9**
14	**12**	**5**	**3**
11	**13**	**4**	**6**
7	**1**	**10**	**16**

u otro cuadrado mágico de 4x4

Apartado A:

- ❑ Implementa la formalización del problema realizada en el TGR1 escribiendo una clase `ProblemaCuadradoMagico` que sea subclase de `ProblemaBusqueda` y define las subclases de `Estado` y `Accion` necesarias para representar el problema.
- ❑ Implementa la estrategia de búsqueda en profundidad y utilízala para resolver el problema (utiliza el primer ejemplo u otras variaciones sencillas).
- ❑ Adapta tu implementación para que lleve cuenta de:
  - ❑ Número de nodos expandidos.
  - ❑ Número de nodos creados.

Apartado B:

- ❑ Indica una heurística apropiada para el problema del cuadrado mágico. Crea una implementación de la clase abstracta `Heuristica` que la calcule para este problema. ¿Es tu heurística admisible? ¿Y consistente?
- ❑ Implementa el método de búsqueda A*. Para ello, crea una subclase de `EstrategiaBusquedaInformada` y modifica la clase `Nodo` para que incluya el coste del camino, el valor de la función *f* y para que implemente el interfaz `Comparable`. Utiliza esta estrategia para resolver el segundo ejemplo.

Apartado C:

- ❑ Implementa el método de búsqueda en profundidad con *backtracking* y utiliza esta estrategia para resolver el tercer ejemplo.
- ❑ ¿Resuelve tu programa el tercer ejemplo en un tiempo razonable?

- **EVALUACIÓN.**

Se realizará una prueba la última sesión de prácticas de laboratorio (del 8 al 14 de marzo de 2023) en el grupo de prácticas correspondiente. La calificación máxima obtenible será de 1,5 puntos.