Laboratorio 2

Ing. Max Alejandro Antonio Cerna Flores

2 de febrero de 2024

1. Descripción

El objetivo de este laboratorio es explorar el algoritmo Graph Search para la resolución de problemas con agentes inteligentes sin heurística y costo de camino.

2. Laboratorio

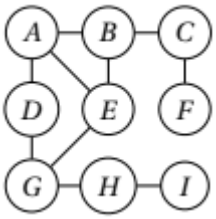
Para iniciar la resolución por favor utilice la siguiente tarea de GitHub Classroom:

https://classroom.github.com/a/i6wl97Vj

En este repositorio de código usted encontrará un proyecto Maven para Java 11, este proyecto está basado en la solución que se le solicitó previamente a este laboratorio en el material complementario. Note que algunos métodos extra en la clase *Node* fueron agregados para su conveniencia.

A partir de acá se le solicitan las siguientes tareas:

1. Modifique la solución para que sea compatible con Java 17
2. En el área de pruebas encontrará la clase *PathFinderTest*, la cual a su vez prueba la funcionalidad de una clase a ser desarrollada denominada *PathFinder*. Esta clase está programada de tal forma que permita resolver un problema de búsqueda representado por el siguiente grafo:



En la firma de los métodos usted observará que esta clase debe ofrecer soporte a búsqueda DFS y BFS. Como primera tarea asegúrese de implementar PathFinder para que su código compile, usted puede decidir la estructura de clases que necesite para este ejercicio siempre y cuando respete PathFinder sin modificar ninguno de los tests.

3. Utilizando el contrato descrito por PathFinder adapte su solución BFS con las siguientes reglas adicionales:

   a) Cada uno de los nodos agregados al espacio explorado debe ser impreso mediante System.out.print mientras la solución es calculada. Por ejemplo si el espacio explorado incluye los nodos X, Y, Z, W en la terminal se debe visualizar como resultado:

   XYZW

   b) En caso de empate, el algoritmo debe dar prioridad a 1- estrategia, 2- orden lexicográfico, 3- aleatoriedad

   c) El resultado del método en PathFinder debe retornar la ruta encontrada por el algoritmo, por ejemplo si X fuera el inicio y W fuera la meta una posible ruta podría ser:

   X->A->B->N->W

4. Utilizando el contrato descrito por PathFinder implemente una solución DFS con las siguientes reglas adicionales:

   a) Cada uno de los nodos agregados al espacio explorado debe ser impreso mediante System.out.print mientras la solución es calculada, por ejemplo si su espacio explorado incluyese los nodos X, Y, Z, W en la terminal se debe ver como resultado XYZW

   b) En caso de empate, el algoritmo debe dar prioridad a 1- estrategia, 2- orden lexicográfico, 3- aleatoriedad

   c) El resultado del método en PathFinder debe retornar la ruta encontrada por el algoritmo, por ejemplo si X fuera el inicio y W fuera la meta una posible ruta podría ser X->A->B->N->W

Fecha límite de entrega: Martes 6 de febrero, 23:59 SIN EXCEPCIONES