Práctica 3
Implementación del problema de las jarras

Fecha de entrega: 14 de noviembre del 2017

El objetivo de esta práctica es implementar un problema de búsqueda usando la librería AIMA.

1. Jungla de Cristal 2

En la película La Jungla de Cristal 2, el malo propone a McCane y a su amigo el siguiente reto[1]: para desactivar la bomba que les ha dejado en una fuente tienen que colocar sobre el maletín que contiene la bomba una garrafa que contenga exactamente 4 galones de agua. Para hacerlo solo disponen de una garrafa de 5 galones, de otra de 3 galones y del agua de la fuente.

2. Solución al problema de Jungla de Cristal 2 con AIMA

Cread un proyecto en Eclipse e importad la librería AIMA que usamos en la Práctica 2 (deberéis de incluir aima-core.jar del paquete release como jar externo en vuestro proyecto).

2.1. Estado del problema

Cread la clase Garrafas.java para representar el estado del problema. Debéis de hacer una representación general que sirva para cualquier capacidad de las jarras iniciales y para cualquier cantidad objetivo.

En esta clase debes de definir la representación de cada estado y las acciones posibles (operadores).

Tendréis que implementar cada uno de los operadores e implementar un método que compruebe si un operador se puede aplicar teniendo en cuenta las precondiciones y postcondiciones.

Deberéis redefinir los métodos equals() y hashCode() para que se ajusten a vuestro problema concreto.

- Como ayuda podéis mirar la clase EightPuzzleBoard que vimos en la Práctica 2.

[1] https://www.youtube.com/watch?v=YuWyTnFVub4

2.2. Comprobación del Estado Objetivo

Cread la clase GarrafasGoalTest que debe implementar la interfaz GoalTest. Esta clase servirá para determinar si un determinado estado es un estado final.

- Como ayuda podéis mirar la clase EightPuzzleGoalTest.java que vimos en la Práctica 2.

2.3. Acciones

Cread la clase GarrafasFunctionFactory para definir el conjunto de acciones posibles que se pueden ejecutar sobre un determinado estado y el cambio de estado que se produce al ejecutar una acción sobre este.

- Como ayuda podéis mirar la clase EightPuzzleFunctionFactory que vimos en la Práctica 2.

2.4. Función heurística

Cread la clase GarrafasHeuristicFunction para definir la heurística que se utilizará en los algoritmos informados. Piensa la idoneidad de la heurística elegida, ¿es admisible? ¿es consistente?

- Como ayuda podéis mirar las clases ManhattanHeuristicFunction y MisplacedTilleHeuristicFunction que vimos en la Práctica 2.

2.5. Demo de aplicación de distintos algoritmos de búsqueda a la resolución del problema

Cread la clase GarrafasDemo que ejecute los siguientes algoritmos de búsqueda para la resolución del problema:

- Búsqueda en anchura con TreeSearch y GraphSearch.
- Búsqueda en profundidad con GraphSearch.
- Búsqueda de coste uniforme con TreeSearch y GraphSearch.
- Búsqueda voraz con GraphSearch.
- Búsqueda A* con TreeSearch y GraphSearch.

Recuerda que la aplicación debe de ser configurable para cualquier estado objetivo. Es decir, en esta clase se deben de configurar los valores de la capacidad de las dos garrafas y la capacidad objetivo.

La ejecución de la clase GarrafasDemo debe de proporcionar para la ejecución de cada uno de los algoritmos, al menos, lo siguiente:

- Traza de ejecución del problema (como va modificándose la cantidad contenida en cada una de las dos garrafas).
- Operadores aplicados.
- Coste de la solución encontrada.
- Número de nodos expandidos.
- Tamaño de la cola.
- Tamaño máximo de la cola.
- Tiempo invertido por el algoritmo en encontrar la solución.

➢ Como ayuda podéis mirar la clase EightPuzzleDemo (aima.gui.demo.search) que vimos en la Práctica 2.

2.6. Estados iniciales a probar

Debéis de ejecutar vuestra aplicación con los siguientes estados iniciales:

1. Capacidad garrafa 1 = 5, capacidad garrafa 2 = 3, cantidad objetivo = 4.
2. Capacidad garrafa 1 = 7, capacidad garrafa 2 = 3, cantidad objetivo = 1.
3. Capacidad garrafa 1 = 12, capacidad garrafa 2 = 3, cantidad objetivo = 1.

3. Memoria

La memoria que debéis de entregar debe de contener el **nombre de los dos integrantes del grupo** y lo siguiente:

1. Tabla con los siguientes datos para cada uno de los estados iniciales y para cada uno de los algoritmos:
   a. Coste de la solución
   b. Nodos expandidos
   c. Tamaño de la cola
   d. Tamaño máximo de la cola
   e. Tiempo de ejecución
2. Análisis de los resultados obtenidos.

4. Entrega de la práctica

La práctica se entregará a través del Campus Virtual. Se habilitará una nueva tarea: **Entrega de la Práctica 3** que permitirá subir un zip que contendrá la memoria de la práctica y las fuentes de la aplicación Java creada.

El fichero subido deberá tener el siguiente nombre: **Practica3GXX.zip**, siendo XX el nombre del grupo. Por ejemplo, *Practica3G01.zip*.

Uno sólo de los miembros del grupo será el encargado de subir la práctica.

La fecha límite para entregar la Práctica 3 será el martes **14 de noviembre a las 23:55**. Se recomienda no dejar la entrega para el último momento para evitar problemas de última hora.

No se corregirá ninguna práctica que no cumpla estos requisitos.