Entrega 3

Jon Acha 79001780T

Sergio Romero 78993882Z

Pregunta 3 - Antes de resolver el problema, ¿sabes qué factor de ramificación "B_level1" va a tener en el nivel 1 de su árbol de búsqueda? y ¿en los niveles 2 y 3 ("B_level2" y "b_level3")?

Nivel 1-7, Nivel 2- 7*6=42 ya que la ciudad visitada no se comprueba, Nivel 3-7*6*5 =210 ya que las ciudades que se van visitando no se podrán visitar y a sí mismo no va a poder ir, de forma que sólo es posible ir a las no visitadas.

Pregunta 4 - Antes de resolver el problema, ¿sabe la profundidad "d" de su árbol de búsqueda?

Será de nivel 7 ya que tiene que visitar todas las ciudades y recorrerlas todas sin repetirlas, en este caso 8 en total si incluimos la ciudad final.

Pregunta 5 - Considera la búsqueda "Best First Search" para resolver el problema. ¿Qué heurística utilizarías? Justifica tu respuesta y garantiza que tu respuesta.

Heurística es ADMISIBLE.

1. Se calcula la distancia de la ciudad actual hasta la final.
2. Por otra parte: Se la ciudad en proceso no se encuentra entre el ArrayList de las ciudades visitadas.
    a. Se calcula la distancia de la ciudad actual a la primera ciudad no visitada con el menor coste posible.
    b. Después de calcularla la ciudad actual pasa a ser la primera ciudad no visitada y así constantemente se vuelve al empezar el paso 2, hasta recorrer todas las ciudades.
3. La heurística que usaríamos sería si distancia entre el camino de la ciudad actual hasta la ciudad final siempre será menor o igual al camino más óptimo en recorrer todas las ciudades visitadas a la ciudad final. Esta heurística siempre será verdadera ya que la distancia de la ciudad actual entre la ciudad final en línea recta es menor o igual de costosa que escoger el camino más óptimo. Esta heurística es admisible ya que la solución siempre va a ser por la ciudad más cercana tomando como referencia la ciudad actual que está en constante cambio.

Analiza el rendimiento de tu sistema con los algoritmos: DEPTH FIRST, BREADTH FIRST y BEST FIRST. El análisis debe incluir:

- Estudio de la susceptibilidad de tu sistema a la ordenación de los operadores en la lista de operadores de tu problema. Explica tus observaciones incluyendo lo robusta que es tu heurística respecto a la ordenación de los operadores. Sugerencia: puedes utilizar el método shuffle() de Collections para reordenar los operadores de forma aleatoria.
- Estudio de la calidad de la solución obtenida (de acuerdo a la medida de rendimiento descrita como parte de la 1 ª entrega) y el tiempo utilizado (o complejidad computacional temporal) por tu sistema; identificando si es posible mejorar alguno de ellos sin comprometer el otro. Si es posible indica cómo.
- En el fichero escribe la posibilidad que ha seleccionado.

DEPTH FIRST

Hemos comprobado que tarda en hallar una solución posible entre 7ms y 9ms realizar el archivo log.

Dada la función shuffle cada ejecución realiza un recorrido distinto si quitamos el shuffle siempre da la misma solución que es el orden: S-G-F-E-D-C-B-A-Z = TOTAL 49.94. Cabe la posibilidad de que el camino elegido no sea el más óptimo dando distancias totales posibles de entre 34-62. Cambiando el orden del fichero XML sin utilizar el método shuffle la solución que saldrá es igual al orden que esta el XML de abajo arriba sin contar la ciudad inicial y la final que siempre irán en el mismo orden.

El fichero log escribe el recorrido realizado por media de los cambios establecidos en la ciudad actual del recorrido escogido aleatoriamente. En el fichero log_solution muestra el recorrido sin la ciudad inicio y la ciudad fin ya que están puestas en otra variable, pero no afecta a la solución ya que siempre serán las mismas.

BREADTH FIRST

Hemos comprobado que tarda en hallar una solución posible entre realizando 1 min 23 seg-2 min el archivo log. Dada la función shuffle cada ejecución realiza un recorrido distinto si quitamos el shuffle siempre da la misma solución que es según el orden del fichero XML de arriba abajo sin contar la ciudad inicial y la ciudad final .

El fichero log escribe el recorrido realizado por medio de los cambios establecidos en la ciudad actual de todos los recorridos que se realizan, con repetición. En el fichero log_solution muestra el recorrido sin la ciudad inicio y la ciudad fin ya que están puestas en otra variable, pero no afecta a la solución.

BEST FIRST

Al ser un método más sofisticado y realizar su tarea mediante una heurística tardará más tiempo entre 13s-14s, Best-First va borrando las ciudades ya visitadas para no tener que calcular la distancia hacia ellas de nuevo.

Dada la función shuffle no afecta a la solución ya que siempre cogerá la misma solución la que para ella es más óptima que siempre tiene una distancia de 32.962. Tampoco se verá afectado por la variación del orden de las ciudades en el documento XML.

En el fichero log se muestran todos los posibles caminos que puede realizar mediante los cambios realizados por medio de la ciudad actual. En el fichero log_solution muestra el recorrido sin la ciudad inicio y la ciudad fin ya que están puestas en otra variables.