Universidad de La Laguna

Diseño y Análisis de Algoritmos

Práctica 4 - Experimentación con Programación dinámica- Problema del Viajante de Comercio (TSP)

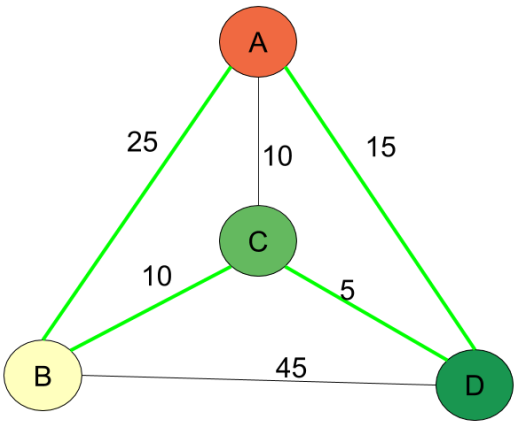
Factor de ponderación: 25%

El problema del viajante de comercio (TSP)

El problema del viajante de comercio o Travelling Salesman Problem (TSP) es un problema NP-Duro altamente conocido en la literatura científica. El problema parte de la existencia de un **grafo completo, no dirigido y pesado**, donde el peso de cada una de sus aristas corresponde a la distancia a viajar entre sus 2 nodos (bidireccional). Existe un comercial que parte de uno de los nodos y para el que se debe obtener la ruta que cumpla lo siguiente:

- Debe pasar por todos los nodos del grafo **exactamente** una vez.
- Debe volver al nodo de origen.
- La distancia total de la ruta debe ser mínima.

Un ejemplo sencillo en un grafo de 4 nodos es el siguiente:



Descripción de la práctica

Para la presente práctica se pide la creación de un software que, **mediante el uso de herencia**, resuelva el TSP de 3 formas diferentes;

- Mediante un **algoritmo de fuerza bruta**. Esto implica probar todas las posibles combinaciones y devolver aquella que retorna mejores resultados.
- Mediante un **algoritmo voraz (no exacto)**. En este algoritmo, se parte de un nodo y en cada momento se elige la ciudad más cercana a la que se pueda desplazar (es decir, por la que no haya pasado ya).
- Mediante un **algoritmo de programación dinámica**. Deben diseñar este algoritmo de cara a la práctica.

El programa debe leer las instancias del problema con un formato específico, donde en la primera línea se indique el número de ciudades disponibles y las siguientes tendrán un formato ***CIUDAD1 CIUDAD2 DISTANCIA***. Así pues, para el ejemplo anterior el fichero de entrada sería el siguiente:

```
4
A B 25
A C 10
A D 15
B C 10
B D 45
C D 5
```

Para cada instancia del problema, el software debe ejecutar los 3 algoritmos implementados, obteniendo de cada ejecución lo siguiente:

- La ruta obtenida con el algoritmo, sobre la cual podremos calcular posteriormente su valor.
- El tiempo de ejecución.

Cabe destacar que al finalizar la ejecución, se debe mostrar una tabla de este tipo (**NOTA**: los datos mostrados son ejemplos):

Instancia	Valor Fuerza Bruta	Tiempo Fuerza Bruta (ms)	Valor Prog. Dinámica	Tiempo Prog. Dinámica (ms)	Valor Voraz	Tiempo Voraz (ms)
4_nodos.txt	55	1578	55	1026	60	50
6_nodos.txt	93	2367	93	1245	100	63

Así pues, el programa debe recibir como parámetro de entrada la ruta de la carpeta donde se encuentran todas las instancias a incluir en este experimento. Para cada una de esas instancias se ejecutarán las 3 versiones del algoritmo y tendrá una fila en la tabla de resultados.

A la ejecución de los algoritmos se le pondrá un tiempo límite de **5 minutos.** En el caso que un algoritmo lleve de ejecución más del tiempo límite, se debe parar dicha ejecución y hacer lo siguiente:

- El resultado devuelto debe ser la mejor ruta obtenida hasta ese momento.
- El tiempo de ejecución en la tabla debe poner el texto **EXCESIVO**.

El tiempo límite de ejecución debe ser fácilmente modificable, ya sea a través de una constante o como un parámetro de llamada al programa.

Además, se debe crear un generador de instancias que, dado un número de nodos, genere una o varias instancias aleatorias para el problema con el número de nodos especificados.

Cada instancia debe escribirse en un fichero con el formato especificado, de tal forma que sea compatible con el programa principal.

Objetivos de la práctica

Los siguientes objetivos se consideran **condición necesaria pero no suficiente** para aprobar la práctica:

1. Crear el software solicitado tal y como se indica en la sección **Descripción de la práctica**.
2. Utilizar el paradigma de **Programación Orientada a Objetos**, así como los lenguajes de programación **C++, C#** o **Java**.

Requisitos evaluables de la práctica

Los siguientes requisitos se evaluarán de cara a la entrega de la práctica:

1. Todo el código deberá estar adecuadamente comentado y desarrollado atendiendo al paradigma de Programación Orientada a Objetos.

2. La arquitectura del software elegida debe ser adecuada a las características requeridas utilizando los principios SOLID.

3. El generador de instancias debe funcionar correctamente y debe ser sencillo de utilizar.

4. Se debe experimentar con instancias de diferente tamaño para poder observar el comportamiento de los algoritmos implementados. Desde instancias de pequeño tamaño a instancias con un gran número de nodos.

Durante la defensa de la práctica **se podrá solicitar algún tipo de modificación o prueba adicional**, la cual afectará en diferente grado a la nota final.

Entrega de la práctica

La práctica debe entregarse en tiempo y forma acorde a lo indicado en la tarea del campus virtual. Para poder considerar la práctica como aprobada, deben cumplirse 2 requisitos:

- La práctica debe defenderse en su sesión correspondiente. Además, debe funcionar tal y como se especifica en el presente enunciado.
- Se debe entregar la tarea del campus virtual incluyendo un fichero con extensión **tar.gz** con el código fuente del programa.
- Se debe entregar, junto al código fuente, un pequeño informe donde se detalle una experimentación con instancias de diferente tamaño.

La no realización de uno de estos 3 puntos conlleva la calificación como suspenso de la práctica.