Parte 2 Análisis de algoritmos Voraces

Problema de las conexiones mínimas (árbol de recubrimiento mínimo)

El Problema de las conexiones mínimas es un problema de optimización combinatoria muy conocido. Dadas una serie de ciudades, el objetivo consiste en interconectar todas las ciudades entre sí con el menor coste total, es decir, partiendo de una ciudad concreta, ver cómo podemos unirla con todas las demás de forma que la distancia total sea mínima.

En su formulación más general, el problema consiste en hallar el árbol de recubrimiento mínimo de un conjunto de nodos: dado una serie de nodos, encontrar el conjunto mínimo de aristas que permita comunicar todos los nodos entre sí recorriendo la menor distancia posible.

Para ello el programa deberá permitir cargar cualquier fichero de la biblioteca TSPLIB (http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/), los cuales presentan el mismo formato, una lista con dos valores para cada ciudad que representan sus coordenadas en el plano. Para componer la matriz de costes se deberá calcular la distancia euclídea entre cada par de ciudades (i, j). Los costes obtenidos han de ser números enteros, es decir, no se considerarán decimales. Así, la matriz de distancias se calcularía en lenguaje C de la siguiente forma:

```
xd = x[i] - x[j];
yd = y[i] - y[j];
dij = rint( sqrt( xd*xd + yd*yd ) );
```

donde `rint` es la función de redondeo y `sqrt` es la raíz cuadrada.

Desarrollo

Se pide al alumno que realice una codificación voraz de los siguientes problemas:

• **berlin52:** Tamaño 52 ciudades. Coste de la solución óptima: *a rellenar por el alumno*
• **ch130:** Tamaño 130 ciudades. Coste de la solución óptima: *a rellenar por el alumno*
• **ch150:** Tamaño 150 ciudades. Coste de la solución óptima: *a rellenar por el alumno*

considerando como criterio voraz la distancia entre las ciudades más cercanas en cada momento:

a) (camino recto más corto) que una alguna de las ciudades ya interconectadas entre sí con alguna no interconectada aún (algoritmos de Prim) o

b) (camino recto más corto) que una dos ciudades entre sí sin provocar un ciclo o circuito (algoritmo de Kruskal)

El alumno deberá entregar un estudio teórico realizado sobre el pseudocódigo simplificado de ambos algoritmos donde se calcule el mejor y el peor caso de manera razonada. Finalmente se pide comparar los resultados esperados del estudio teórico con los experimentales en una tabla y comentar los resultados.

Aplicación Práctica

Para comprobar el correcto funcionamiento del algoritmo voraz implementado el alumno deberá además añadirle al programa una funcionalidad que permita resolver el árbol de recubrimiento mínimo (trazado más corto que permita comunicar todos los nodos entre sí) de cualquier fichero que le indiquemos (el programa debe proporcionar una opción de menú o una ventana de diálogo que permita al usuario indicar la ruta del fichero que desea abrir).

Para ello el programa deberá permitir cargar cualquier fichero de la biblioteca TSPLIB (http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/) cuyo formato sea idéntico al formato del fichero berlin52.tsp.gz existente en el repositorio TSP para problemas simétricos: Symmetric traveling salesman problem (TSP) (http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/tsp/).

El programa deberá calcular cual es el coste de la ruta calculada y deberá mostrar por pantalla el recorrido de dicha ruta, permitiendo guardar en disco un fichero con la solución obtenida.

El formato del fichero generado por la aplicación debe ser el siguiente:

```
NAME : huelva7.opt.tour
TYPE : TOUR
DIMENSION : 7
SOLUTION: 17
TOUR_SECTION
1,2
2,3
4,5
6,7
1,4
4,7
-1
EOF
```