Sexta práctica voluntaria — Métodos Matemáticos y Técn. Comp.
Francisco R. Villatoro y Carmen M. García — 3 de Abril de 2001
PUNTOS : 0.15 — FECHA ENTREGA: Lunes, 23 de Abril de 2001

Programación entera: el problema del viajante.

El problema del viajante: Un representante de comercio parte de la ciudad 0 debiendo realizar un periplo que comprenda las ciudades 1, 2, ..., n, que debe visitar una sola vez, regresando al punto de origen (0). Sea $d_{ij} > 0$ la distancia entre las dos ciudades i y j. El problema consiste en determinar el itinerario de visitas de las ciudades que minimice la distancia total recorrida.

Este problema es de programación lineal entera. Llamemos $n+1$ a la ciudad 0, de forma que queremos visitar las $n+2$ ciudades desde 0 a $n+1$. Introduciremos variables binarias x_{ij}, con $0 \leq i \leq n$ y $1 \leq j \leq n+1$, donde $x_{ij} = 1$ si de la ciudad i se va a la j, y $x_{ij} = 1$ en caso contrario. Nota: $x_{ii} = 0$ siempre.

1. ¿Qué significado tienen las restricciones siguientes?

$$\sum_{i=0}^{n} x_{ij} = 1, \qquad 1 \leq j \leq n+1,$$

$$\sum_{j=1}^{n+1} x_{ij} = 1, \qquad 0 \leq i \leq n.$$

2. Definamos un camino como una sucesión de ciudades

$$t = (0, t_1, t_2, \ldots, t_n, n+1),$$

donde t_k es la ciudad visitada en k-ésimo lugar ($1 \leq t_k \leq n$). Puede ocurrir que un camino incluya bucles ($t_i = t_j$, $i \neq j$), es decir, que se vuelva a la ciudad de partida en menos de $n+1$ pasos. Habrá que evitarlos. Dado un camino t, asociaremos a la ciudad i un valor $\alpha_i = k$, tal que $t_k = i$, es decir, la ciudad i es la k-ésima ciudad visitada en el camino t. La condición

$$\alpha_i - \alpha_j + (n+1)\, x_{ij} \leq n, \qquad 0 \leq i \leq n, \quad 1 \leq j \leq n+1,$$

garantiza la eliminación de todos los bucles (de longitud menor que $n+1$). Demuéstrelo. Ayuda: considere un bucle y sume todas las desigualdades para las ciudades de dicho bucle. Además, demuestre que es suficiente que las α_i sean números reales positivos cualesquiera para que se eliminen los bucles.

3. Formule el problema de programación lineal entera (mixta) que resuelve el problema del viajante: defina la función objetivo, añada los tres tipos de restricciones previamente indicados, las variables α_i serán reales no negativas y las x_{ij} binarias $\{0, 1\}$.

4. Implemente el algoritmo de Ramificación y Acotación (RyA o *Branch&Bound*) para la resolución de un problema de programación entera (mixta) con N variables de las que las últimas M variables son reales.

```
function [sol,flag]=BranchAndBound(A,b,c,M)
%% la función [sol,flag]=BranchAndBound(A,b,c,M)
%% resuelve el problema
%%  min c'x,     S.A.  A.x <= b,
%% donde N=length(c)
%%    x_i es entera para i=1,2, ..., N-M
%%    x_i es real   para i=N-M+1, ..., N
%% y b no está restringida en signo
%%          Mediante el algoritmo Branch & Bound
%% El resultado devuelto es
%% flag= 0, si no existe solución,
%%     = 1, si hay al menos una solución
%%     = inf, si la solución es no acotada
%% sol = solucion, si flag=1
%%     = NaN, en otro caso (flag=0 o =inf).
%%
```

   Para resolver los problemas relajados que aparecen el algoritmo de Ramificación y Acotación utilice la función de Matlab `linprog`, donde `x=linprog(c,A,b)` resuelve el problema

$$\min c' * x, \qquad \text{S.A. } A * x <= b, \quad x \text{ no.rest.sgn.}$$

   ¿Es posible que el algoritmo RyA ramifique dos veces la misma variable en el árbol? En su caso ponga un ejemplo simple.

5. Considere un modelo grosero de España, un cuadrado de lado unidad (en miles de kilómetros). Vamos a generar aleatoriamente N ciudades, las carreteras que las conectan y las distancias entre ellas. Desarrolle un programa de Matlab que genere un problema del viajante (matriz A,

vectores c y b y número M). Dicho programa debe cumplir los siguientes requisitos: que genere aleatoriamente N puntos $(x_i, y_j) \in [0,1]^2$ (serán las N ciudades dentro de dicho cuadrado); utilice `rand` de Matlab; genere un vector aleatorio con el número de carreteras que tiene cada ciudad (al menos dos) y con qué ciudades está conectada (selecciona las ciudades aleatoriamente); calcula las distancias entre cada dos ciudades utilizando la distancia euclídea de los puntos en el cuadrado; elige una ciudad de origen y una de destino.

6. ¿Qué hace la función `line([0;0;1;1;0],[0;1;1;0;0])`? Desarrolle una función para visualizar un mapa de carreteras: dados N puntos en el cuadrado unidad: dibuje el cuadrado unidad, luego dibuje un cuadrado pequeño de lado 0.01 centrado en dicho punto (representará cada ciudad); para cada "carretera"que una dos ciudades dibuje una línea que una los dos puntos que las representan; utilice el mismo color para las ciudades y para las líneas. Ayuda: para dibujar una línea de color rojo utilice

```
gc=line([x1;x2],[y1,y2]); set(gc,'Color','red')
```

7. Resuelva el problema del viajante para un ejemplo con 5 ciudades. Dibuje el mapa de carreteras para dicho ejemplo. Para visualizar la solución obtenida, dibuje la trayectoria del viajante con color rojo. Además añada a cada ciudad un número con su ordinal dentro del camino (ayuda: utilice `text(x,y,'1')`).

8. Resuelva dos problemas con 5 ciudades, uno con 10 y otro con 20. Comente los resultados que ha obtenido. ¿Todo problema tiene solución? Ponga un ejemplo de problema sin solución con 5 ciudades.

9. Vamos a estudiar el coste computacional del algoritmo del viajante. Dibuje una gráfica con el coste medio (en número de operaciones flotantes) en función del número de ciudades del problema. Nota: genere como mínimo 5 problemas para cada número de ciudades y calcule el coste medio. ¿Cómo es la gráfica? ¿El coste es polinomial? ¿Cómo puede saberlo? Justifique sus respuestas.