Modelos Matemáticos en Logística y Transporte
Ejercicio del TSP

Beatriz Coronado Sanz

4 de junio de 2019

Ejercicio. Considere un grafo completo no dirigido con n vértices. Denotemos por c_{ij} el coste de atravesar la arista (i, j) del grafo.

1) Formule como problema de optimización combinatoria la determinación del recorrido que parte del vértice 1, pasa por todos los restantes del grafo y vuelve de nuevo a 1 y tiene coste total mínimo.

Este problema se conoce también como problema del viajante. La forma coloquial de formularlo es: *un viajante de comercio ha de visitar n ciudades, comenzando y finalizando en su propia ciudad. Conociendo el coste de ir de cada ciudad a otra, determinar el recorrido de coste mínimo.*

En teoría de grafos, sea $G = (V, A, C)$ donde $V = \{v_1, ..., v_n\}$ es el conjunto de vértices, $A = \{(v_i, v_j) : v_i, v_j \in V, i \neq j\}$ el conjunto de aristas y $C = (c_{ij})$ la matriz no negativa de costes (c_{ij} es el coste o distancia de la arista (i, j)). El problema del viajante consiste en determinar un tour o ciclo hamiltoniano de coste mínimo.

También podemos formular el problema del viajante mediante un modelo de programación lineal entera con variables binarias (optimización combinatoria). Para ello basta considerar las variables x_{ij} que valen 1 si el viajante va de la ciudad i a la j y 0 en otro caso, y llamar c_{ij} al coste de ir de la ciudad i a la j:

$$
\begin{aligned}
&\text{mín} \sum_{i<j} c_{ij} x_{ij} \\
&\text{s.a.:} \sum_{i=1}^{n} x_{ij} = 1, \ j = 1, 2, ..., n \\
&\sum_{j=1}^{n} x_{ij} = 1, \ i = 1, 2, ..., n \\
&\sum_{(i,j)\in\partial(S)} x_{ij} \geq 2, \ \forall S \subseteq \{1, 2, ..., n\}, \ 3 \leq |S| \leq \left[\frac{n}{2}\right] \\
&x_{ij} \in \{0, 1\} \ \forall i < j
\end{aligned}
$$

donde $\partial(S)$ representa el conjunto de aristas incidentes con exactamente un vértice de S.

Las dos primeras restricciones indican que para cada ciudad solo se llega desde otra ciudad (primera restricción) y se va a otra ciudad (segunda restricción).

Las restricciones que aparecen en tercer lugar (vinculadas a casi la mitad de los subconjuntos de vértices de S) reciben el nombre de restricciones de *eliminación de subtours* y garantizan que la solución sea un tour. Indican que para cada subcojunto de vértices de S con tamaño menor que la mitad del número de vértices, el número de aristas incidentes con un vértice de S debe ser mayor que 2, es decir, que para cada subconjunto de vértices de S debe haber al menos dos aristas que inciden con ese subconjunto: una que entre a él y otra que salga. De esta forma no se generaran subtours en la solución final.

Esta formulación resuelve el problema que queremos porque, dado el ciclo hamiltoniano con mínimo coste que construimos con las variables $x_{ij} = 1$ que obtenemos de resolver el modelo que hemos formulado, podemos considerar el recorrido inducido que parte del vértice 1 y pasa por el resto de los vértices y luego añadir la arista que falta que vuelve al vértice 1 completando el ciclo hamiltoniano.

2) Genere un grafo con costes en las aristas de 10 vértices y resuelva con el solver de su elección el problema anterior sobre este grafo.

Resolver el problema del viajante como se ha formulado anteriormente no es tarea fácil debido a las restricciones de eliminación de subtours, que aumentan exponencialmente a medida que aumenta el número de vértices. Formulado como problema de grafos, existen multitud de solvers que intentan llegar a una solución óptima por medio de algún tipo de heurística, pero no suele existir una versión exacta a la que se pueda acceder.

Para un problema tan pequeño como el que tenemos es factible utilizar una versión exacta que resuelva el problema del viajante, aunque la versión heurística de estos solvers devuelva también una solución del mismo coste en la mayoría de los casos. Debido a esto, se ha programado en R una versión del algoritmo de Held-Karp, que es un algoritmo exacto para el problema del viajante y que lo resuelve por fuerza bruta del mismo modo a como nos pide el enunciado: parte del vértice 1, calcula todos los recorridos posibles utilizando todos los nodos restante y devuelve el ciclo hamiltoniano que, al añadir la arista de vuelta al vértice 1, es mínimo. Esta versión de fuerza bruta es mucho más rápida que una enumeración exhaustiva de soluciones, aunque utiliza también mucho más espacio.

En el archivo *TSP.R* asociado se crea primero un grafo completo de 10 vértices con costes que van de 1 a 100, después se resuelve el problema con el algoritmo de Held-Karp y se da su solución. Por último, aplicamos un solver heurístico con el procedimiento del 2 intercambio y comprobamos que devuelve casi seguro otra solución del problema con el mismo coste.

Por ejemplo, para el grafo que vemos en la siguiente imagen el resultado encontrado es el ciclo hamiltoniano 1 7 3 4 9 6 10 8 5 2 1 y su coste es 176.

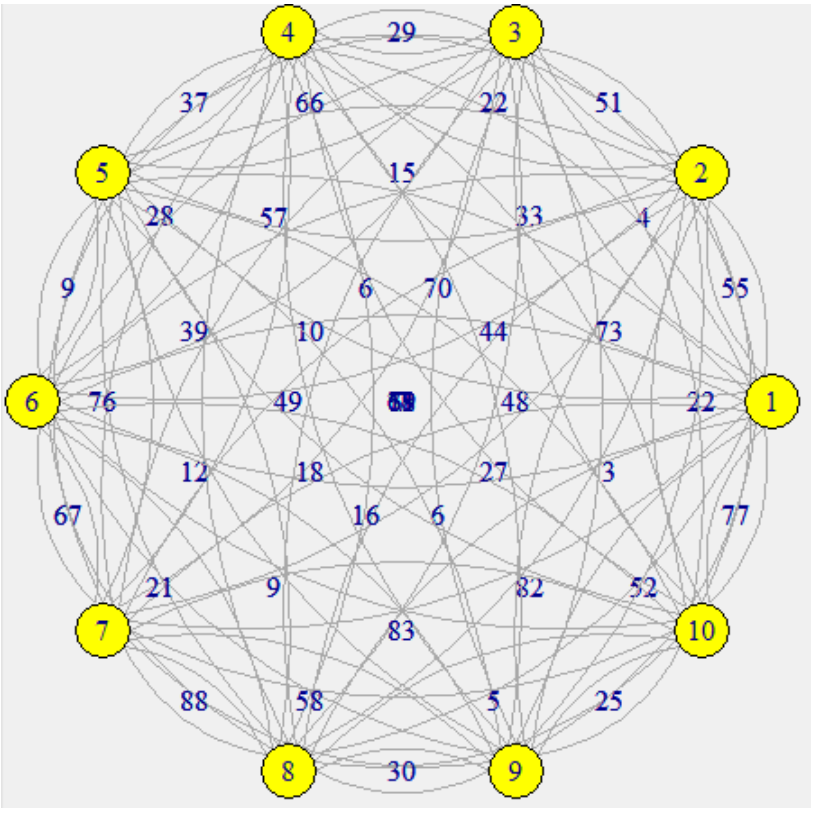

Figura 1: Grafo completo de 10 vértices con costes entre ellos