Tareas Semana I

Jose A. Lozano, Josu Ceberio

8 de octubre de 2019

El objetivo de esta práctica es familiarizarse con la formalización de problemas de optimización combinatoria. Para ello habrá que formalizar dos problemas de optimización combinatoria e implementar el código que permita evaluar una posible solución a cada uno de los problemas una vez dada una instancia.

IMPORTANTE: todas las prácticas de la asignatura se podrán realizar de forma individual o por parejas. Si alguien esta interesado en realizarla por parejas pero no encuentra compañero, por favor, escribid un e-mail al profesor de la asignatura Roberto Santana (roberto.santana@ehu.eus) él tratará de agruparos por parejas. Una vez que se decida hacer esta práctica por parejas el resto de prácticas también se realizarán con la misma pareja y la nota final será la misma para ambos alumnos.

1. El problema de asignación cuadrática

El problema de asignación cuadrática o QAP por sus iniciales en Inglés consiste en asignar un conjunto de localizaciones a un conjunto de instalaciones de manera que se minimize el coste de su utilización. Básicamente el problema viene definido por dos matrices. Una matriz $D = [d_{ij}]$ de dimensión n donde d_{ij} representa la distancia entre las localizaciones i y j, y otra matriz $F = [f_{kl}]$ con la misma dimensión donde f_{kl} representa el flujo entre la instalación k y la instalación l. Si denotamos por $\sigma = (\sigma(1)\sigma(2)\dots\ \sigma(n))$ una posible asignación, la función objetivo se define de la siguiente manera:

$$f(\sigma) = \sum_{i=1}^{n}\sum_{j=1}^{n} f_{ij}d_{\sigma(i)\sigma(j)}$$

2. El problema de la partición del grafo

El problema de la partición de un grafo consiste en lo siguiente: consideremos un grafo no dirigido $G = (\mathcal{X}, U)$ con $\mathcal{X} \neq \emptyset$ y $U = \{(u, v)|u, v \in \mathcal{X}\}$ en el que cada arista (u, v) tiene asociado un peso $p_{(u,v)}$ y el número de vértices es par. Se trata de dividir el conjunto de vértices en dos subconjuntos iguales, de forma que se minimice la suma de los pesos asociados a aristas que unen vértices de diferentes conjuntos.

3. Características del código

Las funciones para evaluar posibles soluciones podrán ser implementados en cualquiera de los dos lenguajes **R** o **Python**. Para realizar la revisión de los algoritmos implementados de manera automática, los programas tendrán nombres predeterminados. La cantidad y el tipo de parámetros de entrada y salida será exactamente los que se describen a continuación independientemente del lenguaje en el que se codifiquen.

Problema de asignación cuadrática

- El programa se llamará: QAPEvaluator
- Parámetros de entrada:
  1. Nombre del fichero con la instancia
  2. Solución a evaluar (permutación de n números, siendo n el tamaño de la instancia)
- Parámetros de salida:
  1. Valor de la función para la instancia.
- El formato de los ficheros conteniendo las instancias será:

  Primera línea: Número de vértices n

  Las siguientes líneas contendrán dos matrices de $n \times n$. La primera matriz es la matriz de pesos y la segunda la matriz de distancias. Se representa una fila de la matriz en cada línea.

  Podeis encontrar ejemplos de ficheros con instancias del problema escritas en este formato en el subdirectorio /Instances/QAP/ incluido dentro de los ejercicios de consolidación.

```
EJEMPLO DE LLAMADA A LA FUNCIÓN EN PYTHON:
  eval = QAPEvaluator('../Instances/QAP/QAP.qap.n10.1',myperm)
```

PROBLEMA QAP

Partición del grafo

- El programa se llamará: BipEvaluator
- Parámetros de entrada:
  1. Nombre del fichero con el grafo
  2. Solución a evaluar (vector binario de tamaño n con exactamente $n/2$ valores igual 1). Notar que n coincide con el número de vértices del grafo de entrada.
- Parámetros de salida:
  1. Valor de la función para la posible solución.
- El formato de los ficheros conteniendo las instancias es el siguiente:

  Primera línea: Número de vértices n

  Las siguientes líneas contendrán una matriz de $n \times n$, una fila de la matriz en cada línea.

  Podeis encontrar ejemplos de ficheros con instancias del problema escritas en este formato en el subdirectorio /Instances/BIPART/ incluido dentro de los ejercicios de consolidación.

  En los ejercicios de consolidación del Tema 2 se incluyen ejemplos de lecturas de instancias utilizando la función Read_Bipart_Instance().

```
EJEMPLO:  edge_weights = Read_Bipart_Instance('../Instances/BIPART/Cebe.bip.n10.1')
EJEMPLO DE LLAMADA A LA FUNCIÓN  EN PYTHON:
  eval = BipEvaluator('../Instances/BIPART/Cebe.bip.n10.1',myvector)
```