ALGORITMIA

Grado en Ingeniería Informática del Software - Curso 2018-2019

GUIÓN DE LA PRÁCTICA 6

OBJETIVO:

- **Algoritmos de backtracking**

Encontrar el emparejamiento estable óptimo

Existen dos conjuntos (uno de n mujeres y otro de n hombres) y tras múltiples estudios, encuestas y entrevistas tenemos como datos de entrada dos matrices cuadradas n*n con la siguiente información:

matHM: cada elemento indica el nivel de rechazo del hombre de la fila correspondiente hacia la mujer de la columna correspondiente. Cada nivel de rechazo toma valores enteros en el rango 0 ..10 (a mayor valor, mayor rechazo).

matMH: cada elemento indica el nivel de rechazo de la mujer de la fila correspondiente hacia el hombre de la columna correspondiente. Cada nivel de rechazo toma valores enteros en el rango 0 ..10 (a mayor valor, mayor rechazo).

Sea por ejemplo el siguiente caso n=6:

matHM=

5	2	6	2	10	8
3	6	0	0	5	10
2	6	0	5	1	10
2	3	5	0	5	1
2	8	9	2	2	8
7	4	3	5	2	7

matMH=

0	2	0	8	2	8
0	3	7	10	1	10
8	7	1	0	7	10
8	4	1	2	7	8
4	7	3	8	1	1
6	10	5	2	9	7

Existe un parámetro fundamental en el sistema llamado **umbralRechazo** (un entero en el rango 0 ..10). Este parámetro se meterá como arg[0] cuando se ejecute la aplicación.

Para que una pareja (hombre, mujer) sea **ESTABLE** ha de cumplirse que el nivel de rechazo de ese hombre hacia esa mujer y el nivel de rechazo de esa mujer respecto a ese hombre sean ambos menores o iguales al parámetro antes descrito umbralRechazo.

Cada pareja ESTABLE la puntuamos con una PUNTUACIÓN igual a la suma de los dos niveles de rechazo mutuos.

Ejercicio pedido

SE PIDE: Diseñar un algoritmo (Clase EmparejamientoEO) capaz de encontrar (si existe) un **EMPAREJAMIENTO ESTABLE ÓPTIMO:**

- **EMPAREJAMIENTO**: implica encontrar n parejas (todo hombre y toda mujer ha de tener pareja).
- **ESTABLE**: Las n parejas han de ser ESTABLES.
- **ÓPTIMO**: La suma de las PUNTUACIONES de esas n parejas (**PUNTUACIÓN TOTAL**) ha de ser mínimo (no puede haber ningún otro emparejamiento estable con puntuación menor).

Caso de prueba:

Si ejecutamos: **java paquete.EmparejamientoEO 5 rechazo06.txt**
(fichero1.txt tiene como contenido el caso antes puesto como ejemplo y un formato análogo al visto en la sesión inicial de prácticas).

Da como resultado:

para umbralRechazo<=3: NO HAY EMPAREJAMIENTO POSIBLE

para umbralRechazo>=4: EMPAREJAMIENTO ÓPTIMO POSIBLE:
hombre[0] --- mujer[1]
hombre[1] --- mujer[3]
hombre[2] --- mujer[2]
hombre[3] --- mujer[5]
hombre[4] --- mujer[0]
hombre[5] --- mujer[4]
PUNTUACIÓN TOTAL MÍNIMA=17

Tiempos de ejecución

Calcular los tiempos de ejecución de la tabla siguiente para un umbralRechazo=5. Las matrices en este caso se generarán aleatoriamente (cada elemento rango 0 .. 10) y para que el tiempo sea más fiable, haga (para cada n) tomas de tiempos sobre 10 matrices diferentes y lleve a la tabla la media aritmética de esos 10 valores temporales.

TABLATIEMPOS:

Caso de estudio (umbralRechazo=5; media de 10 casos aleatorios para cada n)

N	*tiempo*
11	…..
12	…..
13	…..
14	…..
…	…..

Y así sucesivamente hasta que el tiempo se dispare

*Si utiliza **Eclipse**, se creará el proyecto **prac06_Back<UOpropio>** con todas las clases necesarias.*
*Si utiliza **JDK**, cree los paquetes necesarios para esta práctica.*
*Las clases necesarias se crearán dentro del paquete **algdnipropio.p6***

Se entregará, por separado:

- *Los ficheros fuente de las clases, que se hayan programado, dentro del paquete o del proyecto Eclipse.*
- *Un documento con una pequeña explicación de los algoritmos utilizados, su complejidad y la tabla de tiempos de ejecución.*

Se habilitará una tarea en el campus virtual para realizar la entrega. El plazo de entrega será el día anterior al comienzo de la siguiente práctica.