ALGORITMIA

Grado en Ingeniería Informática del Software - Curso 2022-2023

GUIÓN DE LA PRÁCTICA 5

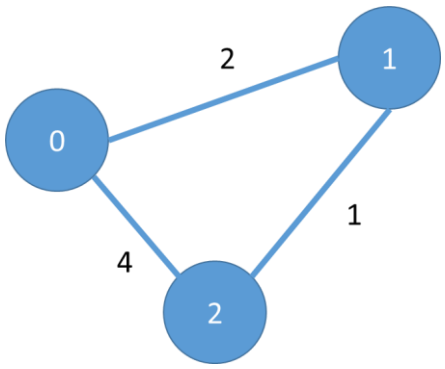
OBJETIVO:

- **Resolución de problemas con algoritmos de programación dinámica**

Camino de anchura máxima entre islas

Un archipiélago consta de unas cuantas islas y varios puentes que unen ciertos pares de islas entre sí. Los puentes construidos tienen ciertas limitaciones en su ancho por diversas razones. Para cada puente (que puede ser de dirección única), además de saber la isla origen y la isla destino, se conoce su anchura > 0. La anchura de un camino formado por una sucesión de puentes es la anchura mínima entre las anchuras de todos los puentes que lo forman. Para cada par de islas se desea saber cuál es el camino de anchura máxima que las une (siempre que exista alguno).

Para ilustrar el planteamiento general véase el ejemplo representado en la siguiente figura:



Aquí tendríamos 3 islas identificadas por los números 0, 1 y 2, todas ellas comunicadas por puentes bidireccionales, el puente entre 0 y 1 tiene una anchura de 2 unidades, el puente entre 0 y 2 una anchura de 4 unidades y el puente entre 1 y 2 una anchura de 1 unidad.

Para dar solución a este caso concreto, podemos ver intuitivamente que para el caso 0 → 1 y 0→2, lo mejor para conseguir el ancho máximo es utilizar los puentes directos, mientras que para el caso 1→2, lo mejor es utilizar un camino indirecto a través de la isla 0, es decir 1→0→2, cuyo ancho será el mínimo de los puentes que hay que atravesar, es decir 2, mejorando el ancho de 1 que tiene el puente directo.

TRABAJO PEDIDO

SE PIDE diseñar e implementar un algoritmo utilizando programación dinámica para resolver este problema de forma óptima.

a) Basándose en ejemplos conocidos similares, diseñar la función recursiva que sirva como base al algoritmo de programación dinámica.
b) Implementar en Java dicho algoritmo de tal forma que calcule, para una configuración de islas, las anchuras óptimas entre cada par de islas y la secuencia de islas que proporciona esa anchura.
c) ¿Qué complejidad tiene el algoritmo diseñado? ¿Crees que se podría cambiar el diseño para mejorar la complejidad?
d) Realice una tabla de tiempos sobre matrices de aristas con pesos de valores aleatorios en el rango [2..10] y con valores del número n de nodos crecientes (n=100, 200, …). ¿Concuerdan los tiempos con lo analizado en el apartado c) anterior?

Los datos de entrada se leerán desde un fichero de texto con el formato especificado

```
3
0 -- 1 2
0 -- 2 4
1 -- 2 1
```

Representando lo siguiente:

- Primera línea: **número de islas**
  - El 3 sería el total de islas
- Segunda y posteriores líneas: **información sobre los puentes que comunican las diferentes islas**
  - Número entero, identificador el nodo origen.
  - Detrás separado por un espacio aparecerá -- si es un puente bidireccional o -> si es dirigido (en una dirección).
  - Número entero, identificador el nodo destino. Separado por un espacio del anterior.
  - Número entero, ancho del puente. Separado por un espacio del anterior.

El programa deberá mostrar los datos leídos originalmente en formato matriz y los resultados finales obtenidos para el fichero dado, incluyendo el array con los anchos óptimos entre cada par de nodos y los recorridos NO directos entre los pares de islas.

Se entregará:

- *Los ficheros fuente de las clases, que se hayan programado, dentro del paquete o del proyecto Eclipse.*
- *Clase de prueba*
- *Un documento PDF con una pequeña explicación del algoritmo utilizado y sus complejidades. El documento también debe incluir las tablas creadas con programación dinámica para los ejemplos pedidos.*
-

Se habilitará una tarea en el campus virtual para realizar la entrega. Esta práctica se trabajará durante una sesión y el ***plazo límite es un día antes*** *de la próxima sesión de prácticas de tu grupo.*