Parcial-2-2022-ENUNCIADO + SOLUC...

EPS_Apuntes

Algoritmia y Estructuras de Datos Avanzadas

2º Grado en Ingeniería Informática

Escuela Politécnica Superior
Universidad Autónoma de Madrid

ENUNCIADO

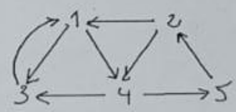
Preguntas

1. a. (2 puntos) i. ¿Cuántos productos de bits hará el algoritmo de Karatsuba para multiplicar dos números de 16 bits?

   ii. Hemos aplicado nuestro algoritmo aproximado para TSP sobre un grafo y nos ha dado un circuito de longitud 1000. ¿Cuál será la longitud mínima de un camino óptimo?

   b. (3 puntos) El algoritmo de Strassen realiza 11 sumas para multiplicar dos matrices 2×2. Expresar en función de N cuántas sumas efecturá para multiplicar dos matrices $N \times N$, siendo N una potencia de 2.

   c. (5 puntos) Encontrar mediante el algoritmo de Tarjan las componentes fuertemente conexas del grafo inferior.



Para ello

- representar las listas de adyacencia de los grafos a considerar;
- representar de alguna manera sobre las mismas la evolución del algoritmo de búsqueda en profundidad;
- representar también cualquier otra información necesaria para el desarrollo del algoritmo como, por ejemplo, bosques de búsqueda en profundidad.

2. a. (2 puntos) i. Si G es un grafo dirigido, definir qué se entiende al decir que es débilmente conexo y al decir que es fuertemente conexo.

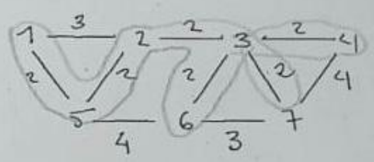
ii. Hemos aplicado búsqueda en profundidad (BP, o DFS) a un grafo dirigido de 5 nodos `1, 2, 3, 4, 5` y obtenido las siguientes tablas de descubrimiento `d` y finalización `f`:

`d = [3, 5, 1, 9, 2] , f = [4, 6, 8, 10, 7] .`

Ordenar los nodos según el orden en el que BP se aplicó a los mismos.

Ordenarlos también según el orden en el que BP finalizó con ellos.

b. (4 puntos) Suponiendo que el grafo inferior es completo y que el coste de las ramas no dibujadas es 4, argumentar que dicho grafo es euclídeo.



A continuación dar una solución aproximada para el problema del viajante sobre dicho grafo detallando los pasos intermedios necesarios y efectuándolos por inspección .

¿Cuál sería un coste mínimo de una solución óptima para dicho problema?

c. (4 puntos) Aplicar los algoritmos QuickSelect y QuickSelect5 a la tabla inferior para buscar su séptimo elemento:

`[11, 3, 10, 2, 6, 9, 7, 1, 8, 5, 4, 12] .`

En QuickSelect tomar como pivote el primer elemento de la sublista con la que se esté trabajando y aplicar la versión simple de `Partir` usada en las slides.

En ambos casos detener el algoritmo cuando la subtabla a explorar tenga 5 o menos elementos.

SOLUCIONES

vodafone yu

FANTASÍA DE FIBRA

SIN PERMANENCIA

yu Fibra

300Mbps

27€ /mes

AEDATA 2022-23

Soluciones de la parte 1 del segundo parcial

a. i. Usando la recursión del algoritmo de Katasuba

$$T(n) = 3T\left(\frac{n}{2}\right)$$

y el hecho que $T(1) = 1$, tenemos

$$T(16) = 3T(8) = 3^2T(4) = 3^3T(2) = 3^4T(1) = 81$$

ii. El algoritmo aproximado nos da un recorrido como mucho el doble del recorrido óptimo. Por otro lado, el óptimo no puede ser más largo que lo que hemos encontrado, por tanto el recorrido óptimo esrtá en el rango $[500, 1000]$

b. No sabemos como funciona el algoritmo de Strassen, pero sabemos que para multiplicar una matrix $n \times n$, en cada paso hace un cierto número K de sumas y siete multiplicaciones de matrices $n/2 \times n/2$. Estas multiplicaciones necesitarán, a su vez sumas y multiplicaciones. Sumar dos matrices $n/2 \times n/2$ necesita $(n/2)^2$ sumas de números reales, por tanto, si $S(n)$ s el número de sumas necesarias para multiplicar dos matrices $n \times n$, podemos escribir

$$S(n) = K\left(\frac{n}{2}\right)^2 + 7S\left(\frac{n}{2}\right)$$

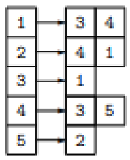
El número K se puede encontrar con los datos del problema. Multiplicar dos matrices 1×1 (es decir, dos números) no necesita ninguna suma, por tanto $S(1) = 0$. Aplicando la fórmula al caso $n = 2$, por que sabemos que se necesitan 11 sumas obtenemos

$$S(2) = K\left(\frac{2}{2}\right)^2 + 7S(1) = K = 11$$

por tanto la iteración es

$$S(n) = 11\left(\frac{n}{2}\right)^2 + 7S\left(\frac{n}{2}\right)$$

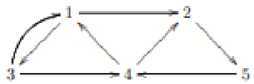
c. La representación del grafo es la siguiente:



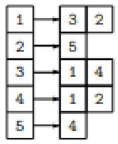
Aplicando el algoritmo dfs empezando, por ejemplo, por el nodo 1, tenemos los siguientes tiempos de visita y funalización

Nodo	$d[u]$	$f[u]$
1	1	10
2	6	7
3	2	3
4	4	9
5	5	8

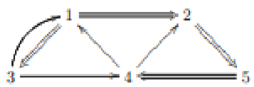
lo que implica que en el dfs sobre la transpuesta del grafo tendrémos que analizar los nodos en el órden $[1, 4, 5, 2, 3]$. La transpuesta del grafo es



con representación



Ejecutando fds encontramos un único árbol:



Lo que implica que el grafo tiene una sola componente conexa, que incluye los nodos $\{1, 2, 3, 4, 5\}$

(Apartado A)

```
   1   2   3   4   5
d  3   5   1   9   2
f  4   6   8   10  7

orden_descubrimiento = [3, 5, 1, 2, 4]
orden_finalización =   [1, 2, 5, 3, 4]
```

(Apartado B)

Se debe cumplir que:
d(u, v) <= d (u, w) + d (w, v) para cualquier terna de nodos u, v, w \in V (pertenecientes al conjunto de nodos de G)
Por ejemplo,
4 <= 2 + 2

Obtener un Árbol Abarcador Mínimo (Minimum Spanning Tree). Por ejemplo

```
   1    2 ----- 3 ------ 4
    \  /       /  \
     5        6    7
```

Doblar aristas del AAM:

```
      1
     ||
     5
     ||
     2
     ||
     3
  // || \\
  6   7  4
```

Obtener un Circuito euleriano (C.E) sobre el grafo anterior:
1 - 5 - 2 - 3 - 4 - 3 - 6 - 3 - 7 - 3 - 2 - 5 - 1

Apartir del C.E anterior ontener un circuito Hamiltoniano (C.H):
1 - 5 - 2 - 3 - 4 - 6 - 7 - 1

Teniendo en cuenta que el coste del circuito hamiltoniano anterior es
c = 2 + 2 + 2 + 2 + 4 + 3 + 4 = 19, el coste mínimo para una solución óptima
c <= 2 c*
c* >= 19/2 = 9.5

(Apartado C)

Teniendo en cuenta que K debe ser el séptimo elemento (k=7) en la ordenación de la tabla

QuickSelect

```
 [3, 10, 2, 6, 9, 7, 1, 8, 5, 4]   11  [12]
        pivot = 11
        index_pivot = 10
        k < (index_pivot + 1)  (7 < 11)

 [2, 1] 3  [10, 6, 9, 7, 8, 5, 4]
        pivot = 3
        index_pivot = 2
        k > (index_pivot + 1),  (7 > 3). Se actualiza el valor de k = 4

 [6, 9, 7, 8, 5, 4]  10
     index_pivote = 6
        k < (index_pivot + 1),  (4 < 7)

 [5, 4] 6 [9, 7, 8]
       pivot = 6
       index_pivot = 2
       k > (index_pivot + 1),  (4 > 3). Se actualiza el valor de k = 1
```

Como el tamaño de la tabla [9, 7, 8] es <= 5 (cutoff), se sale de la recursión,
la tabla se ordena [7, 8, 9] y se devuelve el valor correspondiente a la posición
indicda por k. En este caso la primera posición (K=1)

```
QuickSelect5:

Calculo del pivote: [11, 3, 10, 2, 6,   9, 7, 1, 8, 5,    4, 12]
        medianas = 6, 7, 4 (*)
        Mediana_de_medianas = 6
        pivote = 6

[3, 2, 1, 5, 5 ] 6 [11, 10, 9, 7, 8, 12]
        index_pivot = 5
        k > (index_pivot + 1) (7 > 6). Se actualiza el valor de k = 1

Cálculo del pivote: [11, 10, 9, 7, 8,   12]
        medianas = 9, 12 (*)
        mediana_de_medianas = 9
        pivote = 9

[7, 8] 9 [11, 10, 12]
        pivot = 9
        index_pivot = 2
        k < (index_pivot + 1) (1 < 2)

Cálculo del pivote:

Como el tamaño de la tabla [7, 8] es <= 5 (cutoff), se sale de la
recursión,
la tabla se ordena y se devuelve  el valor correspondiente a la
posición indicda
por k. En este caso primera la posición (K=1)
```