

Ejercicios 3

Análisis de Algoritmos

1. Obtener aproximaciones *tilde* para las siguientes expresiones. Indicar el orden de crecimiento de cada una.

- a) $N+1$
- b) $1+1/N$
- c) $(1+1/N)(1+2/N)$
- d) $2N^3-15N^2+N$
- e) $\lg(2N)/\lg(N)$
- f) $\lg(N^2+1)/\lg(N)$
- g) $N^{100}/2^N$

2. Determinar el orden de crecimiento (en función de N) de los siguientes fragmentos de código. Seleccionar el modelo de costo representativo, estimar la frecuencia del modelo de costo y su orden de crecimiento.

Estructura Unión-Búsqueda

1. Así como se tiene la operación unión(a, b) que conecta dos componentes de la colección, sería posible tener una operación separar(a, b)? Pensar en que sería necesario hacer para implementarla e identificar dificultades para realizar esta operación.

2. Se tienen los elementos 0..9 y se hacen la siguiente secuencia de uniones:

9-0 3-4 5-8 7-2 2-1 5-7 0-3 4-2

- a. Ilustrar la representación al final de esta secuencia que se obtiene aplicando la estructura QuickFind.
- b. Repetir para la estructura QuickUnion. Cuál es la mayor altura?
- c. Repetir para la estructura QuickWeightedUnion. Cuál es la mayor altura?
- d. Cuántas componentes conexas hay al final.

3. El método de compresión de caminos propone conectar todos los nodos a la raíz del árbol para mejorar la eficiencia de las búsquedas. Analizar:

- Como implementar la compresión de caminos al momento de realizar la unión? Cuál sería el orden de crecimiento de la operación unión resultante?
- Como implementar la compresión de caminos al momento de realizar la

búsqueda?Cuál sería el orden de crecimiento de la operación búsqueda resultante?

a.

```
int sum=0;
for(int n=N; n>0; n/=2)
    for(int i=0; i<n; i++)
        sum++;
```

b.

```
int sum=0;
for(int i=1; i<N; i*=2)
    for(int j=0; j<i; j++)
        sum++;
```

c.

```
int sum=0;
for(int i=1; i<N; i*=2)
    for(int j=0; j<N; j++)
        sum++;
```

Estimación de espacio

1. Se declara un arreglo de objetos Fecha:

```
Fecha[] listaCumpleaños = new Fecha[N];  
class Fecha {  
    int anno;  
    byte mes;  
    byte dia;  
}
```

- Estimar el espacio mínimo requerido por el arreglo.
- Estimar el espacio máximo requerido por el arreglo y todas las instancias de Fecha.

2. La siguiente es una implementación de un árbol ternario:

```
class Ternario<T> {  
    Node raíz;  
    class Node {  
        T item;  
        Node izquierdo;  
        Node centro;  
        Node derecho;  
    }  
}
```

Asumiendo que el programa utiliza un Ternario de N items, y que los objetos de tipo T miden K bytes:

- Estimar el espacio requerido por un objeto Node (1 punto)
- Estimar el espacio total requerido por la instancia de Ternario. (1 punto)

Estructura Unión-Búsqueda

1. Así como se tiene la operación unión(a, b) que conecta dos componentes de la colección, sería posible tener una operación separar(a, b)? Pensar en que sería necesario hacer para implementarla e identificar dificultades para realizar esta operación.

2. Se tienen los elementos 0..9 y se hacen la siguiente secuencia de uniones:

9-0 3-4 5-8 7-2 2-1 5-7 0-3 4-2

a. Ilustrar la representación al final de esta secuencia que se obtiene aplicando la estructura QuickFind.

b. Repetir para la estructura QuickUnion. Cuál es la mayor altura?

c. Repetir para la estructura QuickWeightedUnion. Cuál es la mayor altura?

d. Cuántas componentes conexas hay al final?

3. El método de compresión de caminos propone conectar todos los nodos a la raíz del árbol para mejorar la eficiencia de las búsquedas. Analizar:

- Como implementar la compresión de caminos al momento de realizar la unión? Cuál sería el orden de crecimiento de la operación unión resultante?
- Como implementar la compresión de caminos al momento de realizar la búsqueda? Cuál sería el orden de crecimiento de la operación búsqueda resultante?

Respuestas seleccionadas

Análisis de Algoritmos

1

- a) $\sim N$
- b) ~ 1
- c) ~ 1
- d) $\sim 2N^3$
- e) ~ 1
- f) ~ 2
- g) la función tiende a 0 para N muy grande

2

- a. Asumir $N=2^b$.
Ciclo externo se repite para $n= 2^b, 2^{b-1}, \dots, 2^0$, un total de $b+1$ veces.
Ciclo interno se repite para $i=0, \dots, n-1$, para cada valor de n .
Frecuencia de la instrucción `sum++` es
 $2^b + 2^{b-1} + \dots + 2^0 = (2^{b+1}-1)/(2-1)$
- b. El ciclo externo itera para los valores $i=2^0, 2^1, \dots, 2^b$, tales que $2^b < N$. El ciclo interno itera con $j=0, \dots, i-1$ para cada valor de la i .
- c. $(\lg N + 1)N$