(1)

LECCION 1: EFICIENCIA

CASO MEDIO: traza del algoritmo que realiza un nº de instrucciones igual a la esperanza matemática de la variable aleatoria definida por todas las trazas del algoritmo para un tamaño de la entrada, con las posibilidades de que estas ocurran con la entrada.

Ej: ¿Cuál es el nº medio de veces que se ejecuta el while cuando llamo al algoritmo diferentes veces con problemas de tamaño n?

Suponiendo que el dato x tiene la misma probabilidad de que esté en la posición 0, 1, 2, ... n-1 siendo esta probabilidad $\frac{1}{n}$

tenemos que el nº medio de veces que se ejecuta el bucle for es

$$\sum_{i=0}^{n-1} \frac{1}{n} i = \frac{1}{n}\sum_{i=0}^{n-1} i = \frac{1}{n}(0+1+\ldots+n-1)$$

$$= \frac{1}{n}\,\frac{n-1}{2}\cdot n = \frac{n-1}{2}$$

$$T_{1/2}(n) = 1+4+\sum_{j=0}^{\frac{n-1}{2}} 2+4 \quad +3$$

$$= 8+\sum_{j=0}^{\frac{n-1}{2}} 6 = 8+6\left(\frac{n+1}{2}\right) = \boxed{11+3n}$$

Ej: Función que busca la posición máxima de un elemento

```
int Maximo(int a[], int i, int j){
(1) int pmax;
(2) pmax = i;
(3) for (int K = i+1; K <= j; K++)
(4)     if (a[K] > a[pmax])
(5)             pmax = K;
(6) return pmax;
}
```

CASO MEJOR: La condición if siempre es falsa

$$T_m(n) = T(j-i+1) = 1+3+\sum_{K=i+1}^{j}(3+3) \ +1$$

$$= 5+6(j-i)$$

CASO PEOR: La condición if siempre es verdadera.

$$T_p(n) = T(j-i+1) = 1+3+\sum_{K=i+1}^{j}(3+3+1)+1$$

$$= 5+7(j-i)$$

CASO MEDIO: Se puede suponer que la condición IF es verdadera el 50%

$$T_{1/2}(n) = T(j-i+1) = 1+3+\sum_{K=i+1}^{j}\left(3+3+\frac{1}{2}\right)+1$$

$$= 5+\frac{13}{2}(j-i)$$

LECCION : EFICIENCIA

Ej: Ordenación por insercción

IDEA: realiza n-1 iteraciones sobre el vector, dejando en la iesima etapa ($1 \le i \le n-1$) ordenado el vector a[0...i]. Para ello se coloca a[i] en el sitio correcto, teniendo en cuenta que a[0...i-1] ya está ordenado.

```
void Insercción (int a[], int n){
    for (int i=1; i<n; i++){
     int x=a[i]; int j=i-1;
     while (j>=0 && x<a[j]){
          a[j+1]=a[j];
          j=j-1;
     }
     a[j+1]=x;
    }
}
```

CASO MEJOR: El vector esté ordenado. En este caso el while no se ejecuta.

$$T_m(n) = 2 + \sum_{i=1}^{n-1} 2+2+4+1+3+2$$

$$= 14(n-1)+2 = 13n - 11$$

CASO PEOR: Siempre se entra en el while y se llega siempre hasta que $j=0$

$$T_p(n) = 2 + \left(\sum_{i=1}^{n-1} 4+4+\left(\sum_{j=0}^{i-1} 4+4+2\right)+3+2\right)$$

$$= 2 + \sum_{i=1}^{n-1} 13 + \sum_{i=1}^{n-1}\sum_{j=0}^{i-1} 10 =$$

$$= 2 + 13(n-1) + \sum_{i=1}^{n-1} 10i =$$

$$= 2 + 13(n-1) + 10\left(\frac{n}{2}\cdot(n-1)\right)$$

$$= 5n^2 - 5n + 13n - 11 = 5n^2 + 8n - 11$$

CASO MEDIO

Supondremos equiprobable la posición de cada elemento dentro del vector. Por tanto para cada i, la probabilidad de que el elemento se sitúe en la posición K de las i primeras será $\frac{1}{i}$

$$\sum_{j=0}^{i-1} \frac{1}{i}\cdot j = \frac{1}{i}(i-1)\frac{i}{2} = \frac{i-1}{2}$$

$$T_{1/2}(n) = 2 + 13n + \sum_{i=1}^{n-1}\sum_{j=0}^{\frac{i-1}{2}} 10 =$$

$$= 2 + 13n + 10\sum_{i=1}^{n-1}\left(\frac{i+1}{2}\right) =$$

$$= 2 + 13n + 5\left((n+1)\frac{(n-2)}{2}\right)$$

$$= \frac{5}{2}n^2 + 8n - 3$$

③

LECCION 1 : EFICIENCIA

Ordenación por Selección

En cada paso $i = 0 \ldots n-2$ este método busca el mínimo del subvector $a[i..n-1]$ y lo intercambia con el elemento en la posición i:

```
void Seleccion(int a[], int n){
  for (int i=0; i<n-1; i++)
  {
     int pmin=i;
     for (int j=i+1; j<n; j++)
        if (a[pmin]> a[j])
              j=pmin;
     swap(a[i], a[pmin]);
  }
}
```

CASO MEJOR: Nunca se cumple que $a[pmin] > a[j]$

$$T_m(n) = 3 + \sum_{i=0}^{n-2} 1 + 3 + \left(\sum_{j=i+1}^{n-1} (\overset{\text{condicion}}{3} + \underset{j++}{1} + \underset{j<n}{1}) \right) + \overset{\text{swap}}{7}$$

$$= 3 + 11(n-1) + \sum_{i=0}^{n-2} 5(n-i-1)$$

$$= 11n - 8 + \frac{5n^2}{2} - \frac{5n}{2} =$$

$$= \frac{5n^2}{2} + \frac{17n}{2} - 8$$

CASO PEOR: Siempre se ejecuta el if y es true.

$$T_p(n) = 3 + \sum_{i=0}^{n-2} 1 + 3 + \sum_{j=i+1}^{n-1} (3+1+1+1) + 7 =$$

$$= 3 + 11(n-1) + \sum_{i=0}^{n-2} 6(n-i-1)$$

$$= 3n^2 + 8n - 8$$

CASO MEDIO: El if es verdadero el 50% de las veces.

$$T_{1/2}(n) = 3 + \sum_{i=0}^{n-2} 1 + 3 + \sum_{j=i+1}^{n-1} (3+1+1+1/2) + 7 =$$

$$= 3 + 11(n-1) + \sum_{i=0}^{n-2} \frac{5}{2}(n-i-1)$$

$$= \frac{5n^2}{2} + \frac{39}{4} n + 8$$

LECCION 1: EFICIENCIA

COTAS de COMPLEJIDAD: MEDIDAS ASINTÓTICAS

COTA SUPERIOR: NOTACION O

Al conjunto de funciones g que a lo sumo crecen tan deprisa como f se denominan $O(f)$.

Si conocemos la cota superior de un algoritmo siempre aseguraremos que el tiempo de ejecución nunca será peor que esta cota.

-Def-

Sea $f: \mathbb{N} \to [0, \infty)$. Se define el conjunto de funciones de orden $O(f)$ como:

$$O(f) = \{g: \mathbb{N} \to [0,\infty) \mid \exists c \in \mathbb{R}, c>0, \exists n_0 \in \mathbb{N} \text{ tq } g(n) \leq c f(n) \ \forall n \geq n_0\}$$

Ejemplo.

$g(n) = 5n^2 + 9n$

¿cual es el orden de g?

$g(n) \in O(n^2)$ ya que

$$g(n) \leq c \cdot n^2$$

$$5n^2 + 9n \leq cn^2$$

$$cn^2 - 5n^2 - 9n = 0$$

$$n^2(c-5) - 9n = 0$$

$$n(n(c-5) - 9) = 0$$

$$n = \frac{9}{c-5} \quad c > 5 \quad \begin{matrix} c=6 \\ n=9 \end{matrix}$$

PROPIEDADES

1) Para cualquier funcion $f \Rightarrow f \in O(f)$

2) $f \in O(g) \Rightarrow O(f) \subset O(g)$

3) $O(f) = O(g) \Longleftrightarrow f \in O(g) \wedge g \in O(f)$

-Dem-

$\Rightarrow$ $O(f) = O(g)$

Por 1) $f \in O(f) \Rightarrow f \in O(g)$

Por 1) $g \in O(g) \Rightarrow g \in O(f)$

$\Leftarrow$ $f \in O(g) \wedge g \in O(f)$

$$\left.\begin{matrix} f \in O(g) \xrightarrow{2)} O(f) \subset O(g) \\ g \in O(f) \xrightarrow{2)} O(g) \subset O(f) \end{matrix}\right\} O(f) = O(g)$$

4) Si $f \in O(g) \wedge g \in O(h) \Rightarrow f \in O(h)$

-Dem-

$f \in O(g) \Rightarrow f(n) \leq c_1 g(n) \quad \forall n \geq n_0$

$g \in O(h) \Rightarrow g(n) \leq c_2 h(n) \quad \forall n \geq n_1$

$\Downarrow$

$f(n) \leq c_1 \cdot c_2 h(n) \quad \forall n \geq \max(n_0, n_1)$

5) Si $f \in O(g) \wedge f \in O(h) \Longrightarrow$

$f \in O(\min\{g, h\})$

6) REGLA de la SUMA

Si $f_1 \in O(g)$ y $f_2 \in O(h)$

$f_1 + f_2 \in O(\max\{g, h\})$

7) REGLA del Producto

Si $f_1 \in O(g)$ y $f_2 \in O(h)$

$f_1 \cdot f_2 \in O(g \cdot h)$

LECCION 1: EFICIENCIA

8: Si existe $\lim_{n\to\infty} \frac{f(n)}{g(n)} = K$ entonces

1) Si $K \neq 0$ y $K < \infty$ $O(f(n)) = O(g(n))$
2) Si $K = 0$ entonces $f(n) \in O(g(n))$ y $g(n) \notin O(f(n))$
   $\Rightarrow O(f) \subset O(g)$

EJEMPLO

$f(n) = 5n^2 + \log_2(n)$

$g(n) = 8n^2$

$$\lim_{n\to\infty} \frac{5n^2 + \log_2(n)}{8n^2} = \frac{\infty}{\infty}$$

APLICANDO L'HOPITAL

$$\lim_{n\to\infty} \frac{10n + \frac{1}{n}}{16n} = \lim_{n\to\infty} \frac{10n^2+1}{16n^2}$$

$$\simeq \lim_{n\to\infty} \frac{10n^2}{16n^2} = \frac{10}{16} > 0 \Rightarrow O(f(n)) = O(g(n)) = O(n^2)$$

EJERCICIO 2

$f(n) = 16n^3$

$g(n) = 2n^2 + 5n$

¿$O(f) == O(g)$?

COTA INFERIOR. NOTACION Ω

Queremos estudiar aquellas funciones g que a lo sumo crecen tan lentamente como $f \Rightarrow g \in \Omega(f)$

-DEF-

Sea $f: \mathbb{N} \to [0, \infty)$.

$\Omega(f) = \{ g: \mathbb{N} \to [0,\infty) \mid \exists c \in \mathbb{R}, c > 0$
$\exists n_0 \in \mathbb{N}$ tq $g(n) \geq c f(n)$
$\forall n \geq n_0$
$\}$

EJEMPLO

$f(n) = 5n^2 + \log_2(n)$

- $f(n) \in \Omega(n^2)$: es cierto ya que
  $5n^2 + \log_2(n) \geq n^2$ $\forall n \geq n_0 = 1$ y $c = 1$
- $f(n) \in \Omega(\log_2(n))$ es cierto
  $5n^2 + \log_2(n) \geq \log_2(n)$.

Pero entre esas dos cotas escogeremos la mayor que $\Omega(n^2)$.

0'83	0'35	0'8