Estructura de datos - Reto 1

Yábir García Benchakhtir

4 de octubre de 2017

1. Usando la notación O , determinar la eficiencia de los siguientes segmentos de código:

```
1         int n, j; int i=1; int x=0;
2         do{
3           j = 1;
4           while(j <= n){
5             j=j*2;
6             x++;
7           }
8           i++;
9         }while(i <= n)
```

En la primera línea tenemos operaciones de asignación a variables con complejidad $O(1)$.
Encontramos un bucle *do-while* (líneas 2-9) que contiene:

- Una asignación de variable, $O(1)$.
- Un bucle *while* que se repite $log_2 n$ veces con operaciones $O(1)$. La complejidad de esta estructura es pues $log_2(n)$.
- Una operación de incremento que contamos como $O(1)$.

La complejidad de esta última estructura es por tanto $nlog_2 n$ ya que el bucle se repite n veces y la mayor complejidad en terminos de Big O es $log_2 n$. Así la complejidad que obtenemos para el código completo es:

$$O(\text{máx}\{1, nlog_2 n\} = O(nlog_2 n)$$

```
1         int n, j; int i=2; int x=0;
2         do{
3           j = 1;
4           while(j <= i){
5             j=j*2;
6             x++;
7           }
8           i++;
9         }while(i <= n)
```

En el segundo caso estamos repitiendo todo el contenido del *do-while* n veces y dentro de este bucle tenemos una sentencia *while* que depende de una variable j que aumenta de dos en dos. Dentro de este solo tenemos sentencios de complejidad $O(1)$ luego la expresión de la complejidad es del tipo:

$$\sum_{i=1}^{n} log_2(i) = log_2 \prod_{i=1}^{n} i = log_2(n!)$$

2. Para cada función $f(n)$ y cada tiempo t de la tabla siguiente, determinar el mayor tamaño de un problema que puede ser resuelto en un tiempo t (suponiendo que el algoritmo para resolver el problema tarda $f(n)$ microsegundos, es decir, $f(n)^{-6}sg$.

Para ello vamos a obtener una expresión sencilla de la función que la relaciones con el tiempo que se nos pide $t = f(n) \cdot 10^6$.

- $f(n) = log_2$

  Tenemos que despejar el valor de n en la expresión: $10^{-6}log_2 n = t$

  $$10^{-6}log_2 n = t \implies log_2 n = t \cdot 10^6 \implies 2^{t \cdot 10^6} = n$$

- $f(n) = nlog_2 n$

  $$nlog_2 n = t \cdot 10^6$$

  En este caso no existe expresión analitica que podamos simplificar para resolver esta ecuación.

- $f(n) = n$

  $$n = t \cdot 10^6$$

- $f(n) = n^3$

  $$n = \sqrt[3]{t \cdot 10^6} = 100\sqrt[3]{t}$$

- $f(n) = 2^n$

  $$2^n = t \cdot 10^6 \implies n = log_2(t \cdot 10^6)$$

- $f(n) = n!$

  $$n! = t \cdot 10^6$$

Usando las expresiones anteriores despejamos los valores de n para los tiempos dados:

$f(n)$	t				
	1 segundo	1 hora	1 semana	1 año	1000 años
$log_2 n$	$9{,}9 \cdot 10^{301029}$	$2^{3600 \cdot 10^6}$	$2^{6{,}04 \cdot 10^{12}}$	$2^{3{,}15 \cdot 10^{13}}$	$2^{3{,}15 \cdot 10^{16}}$
n	10^6	$3{,}600 \cdot 10^9$	$6{,}048 \cdot 10^{11}$	$3{,}1536 \cdot 10^{13}$	$3{,}1536 \cdot 10^{16}$
$nlog_2 n$	62746	$1{,}33378 \cdot 10^8$	$1{,}77631 \cdot 10^{10}$	$7{,}97634 \cdot 10^{11}$	$6{,}41137 \cdot 10^{14}$
n^3	100	1532	8456	31593	315938
2^n	19	31	39	44	54
$n!$	9	12	14	16	18