UNIVERSIDAD NACIONAL DE COLOMBIA

FACULTAD DE INGENIERIA

ALGORITMOS – Taller 1

EDWIN RICARDO MAHECHA PARRA – 1013679888

1. Ejercicios Cormen '09
   a. 3-12. Demostrar que para cualquier constante real a y b, con b>=0 se cumple que $(n+a)^b = \theta(n^b)$
   Solución:
   Haciendo uso de constantes c y k podemos decir que

$$0 \leq cn^b \leq (n+a)^b \leq kn^b$$
$$0 \leq cn \leq (n+a) \leq kn$$

   Es necesario encontrar valores para c y k en los que se cumpla la desigualdad. Podemos ver que $n + a \leq 2n$ siempre que $abs(a) \leq n$ y también que si $abs(a) \leq \frac{n}{2}$ entonces $(n+2) \geq \frac{n}{2}$. Teniendo en cuenta esto, existen dos constantes $c = \frac{1}{2^b}$ y $k = 2^b$ que satisface la desigualdad para un $n_0 \geq 2abs(a)$
   b. 3-17 Probar que $o(g(n)) \cap \omega(g(n))$ es un conjunto vació.
   Solución:
   Dada las definiciones para o y ω:

$$o\big(g(n)\big) = \{f(n): \forall c, n_0 \; tal \; que \; 0 \leq f(n) \leq \; cg(n), \forall n \; > n_0\}$$
$$\omega\big(g(n)\big) = \{f(n): \forall c, n_0 \; tal \; que \; 0 \leq cg(n) \leq \; f(n), \forall n \; > n_0\}$$

   Podemos definir la intersección de ambas como

$$o(g(n)) \; y \; \omega(g(n)) \; = \; \{f(n): \forall c, n_0 \; \leq cg(n) \; \leq \; f(n) \; \leq cg(n), \forall n \; > n_0\}$$

   Dicha desigualdad no se cumple para ningún $f(n)$ por lo que el conjunto es vació.
   c. 3-3 a) Rank the following functions by order of growth; that is, find an arrangement g1, g2, …. , g30 of the functions satisfying $g1 = \Omega(g2), \; g2 = \Omega(g3), \ldots, g29 = \Omega(g30)$.Partition your list into equivalence classes such that functions f(n) and g(n) are in the same class if and only if $f(n) = \Theta(g(n))$.
   Solución:

1	$2^{2^{n+1}}$		13	$\lg(n!)$	$nlg(n)$
2	2^{2^n}		14	n	$2^{\lg(n)}$
3	$(n+1)!$		15	$\sqrt{2}^{\lg(n)}$	
4	$n!$		16	$2^{\sqrt{2\lg(n)}}$	
5	$n2^n$		17	$lg^2(n)$	
6	e^n		18	$\ln(n)$	
7	2^n		19	$\sqrt{\lg(n)}$	
8	$\frac{3^n}{2^n}$		20	$lnln(n)$	
9	$(\lg(n))!$		21	$2^{lg^* n}$	

10	$(\lg(n))^{\lg(n)}$	$n^{\lg(\lg(n))}$	22	$lg^*(n)$	$lg^*(\lg(n))$
11	n^3		23	$\lg(\lg^*(n))$	
12	n^2	$4^{\lg(n)}$	24	$\frac{1}{n^{\lg(n)}}$	1

b) Give an example of a single nonnegative function f(n) such that for all functions gi (n) in part (a), f(n) is neither O(gi (n)) nor $\Omega(gi\ (n))$.

Solución

Definiendo la función:

$$f(n) = \begin{cases} g_1(n)! & n \mod 2 = 0 \\ \frac{1}{n} & n \mod 2 = 1 \end{cases}$$

Y teniendo en cuenta que f(n) es positiva. Para n par se tiene que:

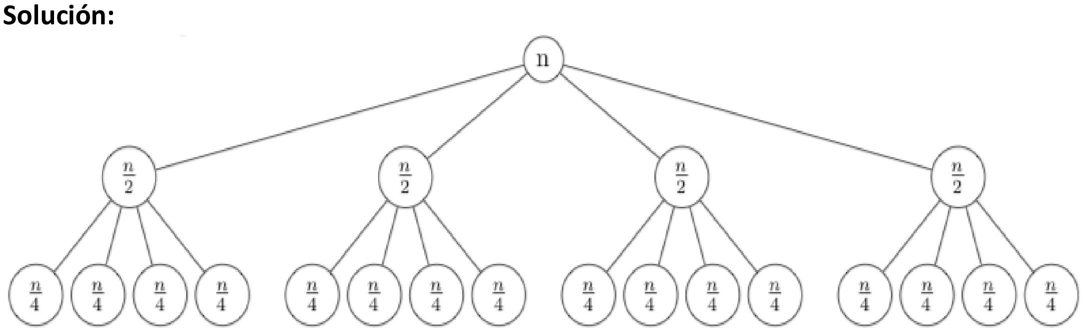
$$\begin{aligned} \lim_{n\to\infty} \frac{f(2n)}{g_i(2n)} &\geq \lim_{n\to\infty} \frac{f(2n)}{g_1(2n)} \\ &= \lim_{n\to\infty} (g_1(2n) - 1)! \\ &= \infty \end{aligned}$$

Y para impar:

$$\begin{aligned} \lim_{n\to\infty} \frac{f(2n+1)}{g_i(2n+1)} &\leq \lim_{n\to\infty} \frac{f(2n+1)}{1} \\ &= \lim_{n\to\infty} \frac{1}{2n+1} \\ &= 0 \end{aligned}$$

d. 4-47 Draw the recursion tree for T(n) = 4T([n/2]) + cn, where c is a constant, and provide a tight asymptotic bound on its solution. Verify your bound by the substitution method.

Solución:



Cada nodo del árbol incrementa el numero de subproblemas en 4 y cada subproblema tiene un coste en tiempo de n/2.

Resolviendo el árbol, obtenemos que el costo total del árbol es de:

$$\begin{aligned}
T(n) &= \sum_{i=0}^{\lg n} 4^i \cdot c\left(\frac{n}{2^i}+2\right) \\
&= \sum_{i=0}^{\lg n} 4^i \cdot c\frac{n}{2^i} + \sum_{i=0}^{\lg n} 4^i \cdot c \cdot 2 \\
&= cn\sum_{i=0}^{\lg n} \frac{4^i}{2^i} + 2c\sum_{i=0}^{\lg n} 4^i \\
&= cn\sum_{i=0}^{\lg n} 2^i + 2c\sum_{i=0}^{\lg n} 4^i \\
&= cn\frac{2^{\lg n+1}-1}{2-1} + 2c\frac{4^{\lg n+1}-1}{4-1} \\
&= cn(2^{\lg n+1}-1) + \frac{2c}{3}(4^{\lg n+1}-1) \\
&= cn(2\cdot 2^{\lg n}-1) + \frac{2c}{3}(4\cdot 4^{\lg n}-1) \\
&= cn(2\cdot n-1) + \frac{2c}{3}(4\cdot n^2-1) \\
&= 2cn^2 - 2n + \frac{8cn^2}{3} - \frac{2c}{3} \\
&= O(n^2)
\end{aligned}$$

Que sustituyendo de forma iterada queda[1]:

$$\begin{aligned}
T(n) &= 4T(n/2+2)+n \\
&\le 4d((n/2+2)^2 - b(n/2+2)) + n \\
&= 4d(n^2/4 + 2n + 4 - bn/2 - 2b) + n \\
&= dn^2 + 8dn + 16d - 2dbn - 8db + n \\
&= dn^2 - dbn + 8dn + 16d - dbn - 8db + n \\
&= d(n^2 - bn) - (db - 8d - 1)n - (b-2)8d \\
&\le d(n^2 - bn)
\end{aligned}$$

e. Use el método maestro para dar cotas ajustadas para las siguientes recurrencias:

- $T(n) = 8T\left(\frac{n}{2}\right) + n$

  $$n^{log_2 8} = n^3 = \theta(n^3)$$

  f(n) tiene una cota $\theta(n^{log_b(a)-\epsilon})$ y $\epsilon = 2$ por lo que $T(n) = \theta(n^3)$

- $T(n) = 8T\left(\frac{n}{2}\right) + n^3$

  $$n^{log_2 8} = n^3 = \theta(n^3)$$

  f(n) tiene una cota $\theta(n^{log_b(a)})$ por lo que $T(n) = \theta((n^3)\log(n))$

- $T(n) = 8T\left(\frac{n}{2}\right) + n^5$

  $$n^{log_2 8} = n^3 = \theta(n^3)$$

  f(n) tiene una cota $\theta\left(n^{log_b(a)-\epsilon}\right)$ y $\epsilon = 2$ por lo que $T(n) = \theta(n^5)$

2. Dado el pseudocodigo:

[1] Se cumple para $db - 1 - 8d \ge 0$

```
def misterio(n):
   if n <= 1:
       return 1
   else:
       r = misterio(n / 2)
       i = 1
       while n > i*i:
           i = i + 1
       r = r + misterio(n / 2)
       return r
```

a. Plantee una ecuación de recurrencia para $T(n)$, el tiempo que toma la función misterio(n).
   Solución:
$$2T\left(\frac{n}{2}\right) + \theta(\sqrt{n})$$
b. Dibuje el árbol de recursión y calcule:
   - Altura del árbol: $log_2(n)$ para el nivel n-ésimo.
   - # de nodos por cada nivel:$2^{\log(n)}$
   - Suma de nodos de cada nivel: $2^i k(\frac{n}{2^i})^{1/2}$
   - Suma total: $\theta(n)$
c. Determine el comportamiento asintótico de $T(n)$ justificándolo de manera detallada.
   Solución:
   Haciendo uso del método maestro tenemos que: a = 2, b = 2 y f(n) es de la forma$O(n^{log_b(a)-\epsilon})$ con $\epsilon = 0.5$ por lo que $T(n) = \theta(n)$.

3. Cormen '09 22.3-1 Realizar un cuadro 3x3 con filas y columnas WHITE, GRAY, BLACK. En cada celda (i,j), indique si en algún momento durante la búsqueda en la profundidad de un grafo dirigido, puede haber una arista o un vértice de color i a un vértice de color j. Por cada arista posible indique que tipo puede ser. Haga otro cuadro, pero utilizando un grafo no dirigido.
   Solución:
   Para el grafo dirigido:

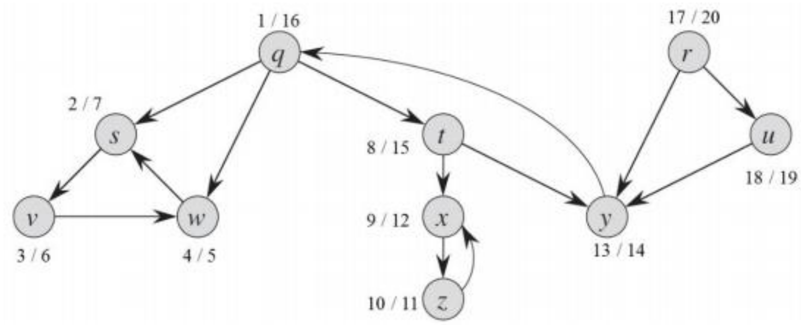
i/j	White	Gray	Black
White	Todos	Cross, back	Tree, cross, forward
Gray	Back, cross	Tree, forward, back	Tree, forward, cross
Black	Back, cross	Back, cross	Todos

   Para el grafo no dirigido:

i/j	White	Gray	Black
White	Todos	-	-
Gray	Todos	Tree, forward, back	-
Black	Todos	Todos	Todos

4. Cormen '09 22.3-2 Muestre como DFS funciona en el grafo de la figura 22.6. Asuma que el ciclo for de la línea 5-7 del DFS considera los vértices en orden alfabético, y asuma que

cada lista de adyacencia está ordenada alfabéticamente. Muestre el descubrimiento y los tiempos de cada vértice y muestre la clasificación de cada arista.



a. Forward edges: (q,w).
b. Back edges: (z,x), (w,s), (y,q).
c. Tree edges: (q,s),(s,v),(v,w),(q,t),(t,x),(x,z),(t,y),(r,u).
d. Cross edges: (u,y),(r,y)

5. Cormen ’09 22.4-2 Dé un algoritmo de tiempo lineal que tome como input un grafo dirigido acíclico $G = (V, E)$ y dos vértices s y t, y retorna el número de caminos simples de s a t en G.

```
if u == v then
     Return 1
else if u.paths 6= NIL then
     Return u.paths
else
     for each w ∈ Adj[u]
          u.paths = u.paths+ SIMPLE-PATHS(w, v)
     Return u.paths
```