UNIVERSIDAD NACIONAL DE COLOMBIA

FACULTAD DE INGENIERIA

ALGORITMOS – Taller 1

EDWIN RICARDO MAHECHA PARRA – 1013679888

1. Ejercicios Cormen ‘09
   1. 3-12. Demostrar que para cualquier constante real a y b, con b>=0 se cumple que   
      **Solución:**

Haciendo uso de constantes c y k podemos decir que

Es necesario encontrar valores para c y k en los que se cumpla la desigualdad. Podemos ver que siempre que y también que si entonces . Teniendo en cuenta esto, existen dos constantes y que satisface la desigualdad para un

* 1. 3-17 Probar que es un conjunto vació.

**Solución:**

Dada las definiciones para y :

Podemos definir la intersección de ambas como

Dicha desigualdad no se cumple para ningún por lo que el conjunto es vació.

* 1. 3-3 a) Rank the following functions by order of growth; that is, ﬁnd an arrangement g1, g2, …. , g30 of the functions satisfying .Partition your list into equivalence classes such that functions f(n) and g(n) are in the same class if and only if .

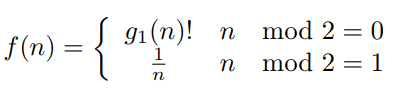
**Solución:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 |  |  | 13 |  |  |
| 2 |  |  | 14 |  |  |
| 3 |  |  | 15 |  |  |
| 4 |  |  | 16 |  |  |
| 5 |  |  | 17 |  |  |
| 6 |  |  | 18 |  |  |
| 7 |  |  | 19 |  |  |
| 8 |  |  | 20 |  |  |
| 9 |  |  | 21 |  |  |
| 10 |  |  | 22 |  |  |
| 11 |  |  | 23 |  |  |
| 12 |  |  | 24 |  |  |

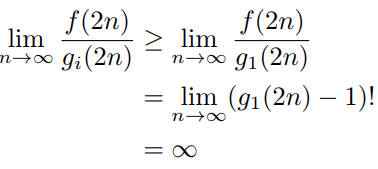
b) Give an example of a single nonnegative function f(n) such that for all functions gi (n) in part (a), f(n) is neither O(gi (n)) nor .

**Solución**

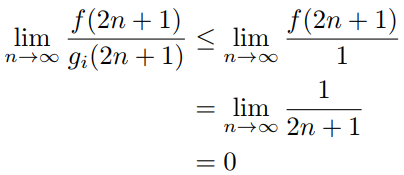
Definiendo la función:



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

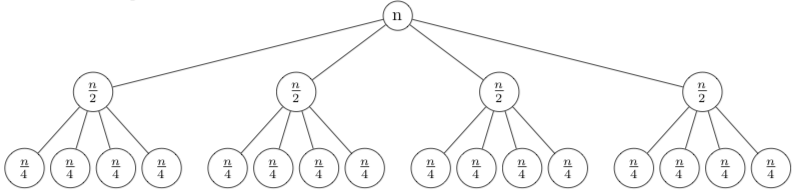


Y para impar:



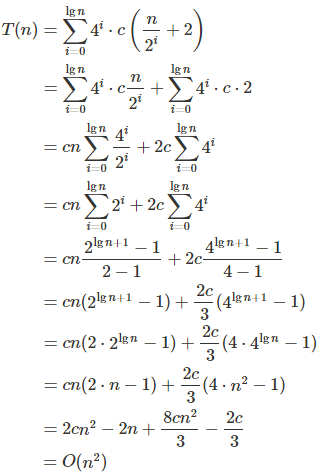
* 1. 4-47 Draw the recursion tree for T(n) = 4T([n/2]) + cn, where c is a constant, and provide a tight asymptotic bound on its solution. Verify your bound by the substitution method.

**Solución:**

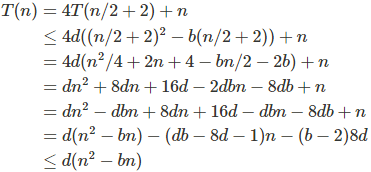
****

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:



Que sustituyendo de forma iterada queda[[1]](#footnote-1):



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

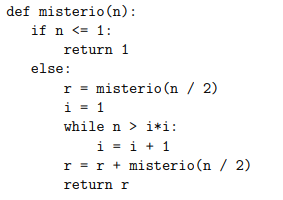
f(n) tiene una cota y por lo que

f(n) tiene una cota por lo que



f(n) tiene una cota y por lo que

1. Dado el pseudocodigo:



* 1. Plantee una ecuación de recurrencia para , el tiempo que toma la función misterio(n).

**Solución:**

* 1. Dibuje el árbol de recursión y calcule:
     + Altura del árbol: para el nivel n-ésimo.
     + # de nodos por cada nivel:
     + Suma de nodos de cada nivel:
     + Suma total:
  2. Determine el comportamiento asintótico de justificándolo de manera detallada.

**Solución:**

Haciendo uso del método maestro tenemos que: a = 2, b = 2 y f(n) es de la forma con por lo que .

1. 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:**

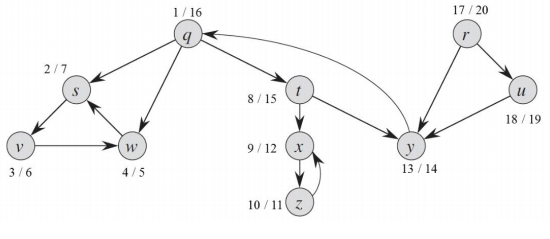
Para el grafo dirigido:

|  |  |  |  |
| --- | --- | --- | --- |
| 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 |

1. 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.



* 1. Forward edges: (q,w).
  2. Back edges: (z,x), (w,s), (y,q).
  3. Tree edges: (q,s),(s,v),(v,w),(q,t),(t,x),(x,z),(t,y),(r,u).
  4. Cross edges: (u,y),(r,y)

1. 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

1. Se cumple para [↑](#footnote-ref-1)