Tarea 01 Árboles y Grafos

Josue Peña Atencio - 8935601

3 de agosto de 2018

Índice

1. Código de honor

Como miembro de la comunidad académica de la Pontificia Universidad Javeriana Cali me comprometo a seguir los más altos estándares de integridad académica.

2. Ejercicios 1, 3 y 5

2.1. Ejercicio 1

1. *a)* $(2n)^2 = 4n^2$, se hace 4 veces más lento.

$(n+1)^2 = n^2 + 2n + 1$, se hace mas lento por una adición de $2n + 1$.

b) $(2n)^3 = 8n^2$, se hace 8 veces más lento.
$(n+1)^3 = n^3 + 3n^2 + 3n + 1$, se hace mas lento por una adición de $3n^2 + 3n + 1$.

c) $(100(2n))^2 = 4(100n^2)$, se hace 4 veces más lento.
$100(n+1)^2 = 100n^2 + 200n + 100$, se hace mas lento por una adición de $200n + 100$.

d) $(2n)\log(2n) = 2(n \log n) + 2n(\log 2)$, se hace 2 veces más lento más una adición de $2n(\log 2)$.
$(n+1)\log(n+1) = n\log n + \log(n+1) + n[\log n + 1 - \log n]$, se hace más lento por una adición de $\log(n+1) + n[\log n + 1 - \log n]$.

e) $2^{(2n)} = (2^n)^2$, se hace dos ordenes de magnitud más lento.
$2^{(n+1)} = 2(2^n)$, se hace 2 veces más lento.

2.2. Ejercicio 3

$$\sqrt{2n} \leq n + 10 \leq n^{2,5} \leq n^2 \log n \leq 10^n \leq 100^n$$

2.3. Ejercicio 5

1. Si tomamos $n_0 = 2$, tal que $n \geq n_0$ para todo n, la declaración es verdadera, ya que como $f(n) \leq cg(n) \Rightarrow \log_2 f(n) \leq \log_2 g(n) + \log_2 c \leq (\log_2 c)(\log_2 g(n))$.

2. Falso. Ya que si $f(n) = 2n$ y $g(n) = n + 1$, tal que $f(n) \leq 2g(n)$, con $n = 2$, tenemos $2^{(2n)} \leq 2(2^{n+1}) = 2^{n+2}$ lo cual es falso para cualquier $n \geq 2$.

3. Verdadero. Ya que como $f(n) \leq cg(n) \Rightarrow (f(n))^2 \leq c^2(g(n))^2$. La desigualdad se mantiene para cualquier $n \geq n_0$.

3. Ejercicio 8: *Safe rungs*

1. *a*) Ya que el algoritmo para resolver este problema requiere una complejidad menor que lineal, y el algoritmo de complejidad logarítmica quiebra más frascos que el lineal, el mejor candidato es un algoritmo de complejidad $O(n^{\frac{1}{2}})$.

   Es más barato aumentar la altura de los escalones de $n^{\frac{1}{2}}$ en $n^{\frac{1}{2}}$, así cuando el frasco se quiebre al caer de un escalón, podremos ir subiendo la altura de lanzamiento a partir del escalón del lanzamiento anterior, ya no en factores de $n^{\frac{1}{2}}$ sino en unidades (de 1 en 1), para poder encontrar el escalón seguro más alto.

   b)

4. Ejercicio 4: *Flipping Pancakes*

1. *a*) Para poder ordenar una pila de n pancakes se deben hacer los siguientes pasos:
   1) Buscar el pancake más grande que no esté ordenado
   2) Insertar la espátula debajo del pancake encontrado y hacer un *flip* de la pila de modo que el pancake quede en la cima
   3) Voltear la pila completa desde el ultimo pancake ordenado de manera que el que estaba en la cima ahora quede encima del ultimo ordenado, ordenándolo en el proceso.
   4) Repetir hasta que toda la pila esté ordenada

   El peor caso sería en el cual para cada pancake de la pila se deban realizar los dos *flips* descritos en el algoritmo. Dado que la pila es de tamaño n, se realizarían $2n$ *flips* en el peor caso.

2. En este caso para los pancakes quemados, el algoritmo es exactamente el mismo que el anterior, solo que con una pequeña variación.

   Se realizan los mismos pasos, pero cuando el pancake más grande no ordenado ha sido puesto en la cima de la pila, si este tiene el lado quemado hacia abajo, se debe poner la espátula justo debajo de el, y darle la vuelta, de modo que su lado quemado quede hacia arriba. Luego, se procede a voltear toda la pila a partir del ultimo pancake ordenado, de esa forma el pancake a punto de ser ordenado quedará con su lado quemado hacia abajo.

   Justo como en el caso anterior, el peor caso será en el cual se deban hacer la cantidad de *flips* descrita en el algoritmo para cada uno de los pancakes. Por lo cual en este caso la máxima cantidad de *flips* es $3n$.

5. Ejercicio 14: *Inversions*

Este algoritmo devuelve la cantidad de inversiones en un arreglo en tiempo $O(n \log n)$. Contar las inversiones en un arreglo es lo mismo que contar la cantidad mínima de *flips* que se deben hacer entre elementos adyacentes para arreglar el mismo.

Algoritmo sacado de la clase del 25 de Julio:

```
def f(arr, lo, hi):
  if hi - lo <= 1:
    return 0

  mid = (lo + hi) // 2
  ans = f(arr, lo, mid)
  ans = ans + f(arr, mid, hi)
```

```
aux = []
i = lo
j = mid
while i < mid and j < hi:
  if arr[i] <= arr[j]:
    aux.append(arr[i])
    i = i + 1
  else:
    aux.append(arr[j])
    ans = ans + (mid - i)
    j = j + 1

while i < mid:
  aux.append(arr[i])
  i = i + 1

while j < hi:
  aux.append(arr[j])
  j = j + 1

arr[lo..hi] = aux
return ans
```

Si un elemento *arr[i]* en un segmento ordenado del arreglo es mayor que otro *arr[j]* en el otro segmento del arreglo también ordenado, entonces todos los elementos *arr[i .. mid]* también son mayores que *arr[j]*, por lo cual en ese caso hay *mid* - *i* inversiones.

6. Ejercicios 4a, 4b y 13

Pruebas por inducción.

6.1. Ejercicio 4a

Probar

$$\sum_{i=0}^{n} F_i = F_{n+2} - 1$$

(H.I)

Caso base

$$\begin{aligned} n &= 0 \\ F_0 &= F_2 - 1 \\ F_0 &= (F_1 + F_0) - 1 \\ 0 &= (1 + 0) - 1 \\ 0 &= 0 \end{aligned}$$

Caso inductivo

$$\sum_{i=0}^{n+1} F_i = F_{n+3} - 1$$

$$\sum_{i=0}^{n} F_i + F_{n+1} = F_{n+3} - 1$$

Por hipótesis de inducción tenemos

$$F_{n+2} - 1 + F_{n+1} = F_{n+3} - 1$$

Por definición de números Fibonacci tenemos

$$F_{n+3} - 1 = F_{n+3} - 1 \blacksquare$$

6.2. Ejercicio 4b

Probar

$$F_n^2 - F_{n+1}F_{n-1} = (-1)^{n+1}$$

(H.I)

Caso base

$$\begin{aligned} n &= 1 \\ F_1^2 - F_1F_0 &= (-1)^2 \\ 1^2 &= (-1)^2 \\ 1 &= 1 \end{aligned}$$

Caso inductivo

$$F_{n+1}^2 - F_{n+2}F_n = (-1)^{n+2}$$

Por definición de números Fibonacci tenemos

$$F_{n+1}^2 - (F_{n+1} + F_n)F_n = (-1)^{n+2}$$
$$F_{n+1}^2 - (F_{n+1}F_n + F_n^2) = (-1)^{n+2}$$

Por hipótesis de inducción tenemos

$$F_{n+1}^2 - (F_{n+1}F_n + (-1)^{n+1} + F_{n+1}F_{n-1}) = (-1)^{n+2}$$
$$F_{n+1}^2 - (F_{n+1}(F_n + F_{n-1}) + (-1)^{n+1}) = (-1)^{n+2}$$

Por definición de números Fibonacci tenemos

$$F_{n+1}^2 - (F_{n+1}(F_{n+1}) + (-1)^{n+1}) = (-1)^{n+2}$$
$$F_{n+1}^2 - F_{n+1}^2 + (-1)^{n+2} = (-1)^{n+2}$$
$$(-1)^{n+2} = (-1)^{n+2} \blacksquare$$

6.3. Ejercicio 13

Caminos de Hamilton en grafos Torneos

n: cantidad de vértices del grafo tipo Torneo.

Probar Todo grafo tipo Torneo tiene un camino de Hamilton.

Caso base $n = 1$. Cuando solamente hay un solo vértice, el camino de Hamilton claramente existe, este es él mismo.

Hipótesis de inducción Ahora, suponemos que un grafo Torneo de tamaño n tiene un camino de Hamilton, tal que la secuencia $v_1, v_2, v_3, ...v_n$ denota el orden en el que se deben recorrer los vértices para realizar todo el recorrido del camino.

Caso inductivo Consideremos ahora un grafo con n + 1 vértices, del cual removemos un vértice v cualquiera junto con todas las aristas entrantes y salientes de éste.

Por nuestra hipótesis de inducción, este nuevo grafo de n vértices tiene un camino de Hamilton.

Para añadir entonces el vértice arbitrario v al grafo de nuevo de forma que el resultado sea otro grafo con camino de hamilton, podemos considerar tres casos diferentes para hacer esto.

Caso 1 Si tenemos la secuencia $v_1, v_2, v_3, ...v_n$ denotando el camino de Hamilton del grafo antes de la inserción del nuevo vértice v, si creamos la arista $(v \to v_1)$, la secuencia del camino ahora sería $v, v_1, v_2, v_3, ...v_n$

Caso 2 Similar a la primer forma, si creamos una arista $(v_n \to v)$, la secuencia del recorrido sería $v, v_1, v_2, v_3, ...v_n, v$

Caso 3 De no poder realizar la inserción del nuevo vértice de las dos formas anteriores, si tenemos una arista cualquiera en la secuencia del camino v_i podemos crear dos aristas $(v_i \to v)$ y $(v \to v_{i+1})$ de tal forma que la secuencia final del camino de Hamilton sea $v_1, v_2, v_3, ...v_i, v, v_{i+1}, ...v_n$

Usando cualquiera de las tres estrategias siempre podremos generar un nuevo grafo tipo Torneo que tenga un camino de Hamilton. ■

7. Referencias

7.1. Material consultado

1. `https://en.wikipedia.org/wiki/Pancake_sorting`
2. `https://en.wikipedia.org/wiki/Tournament_(graph_theory)`
3. `http://www.math.toronto.edu/gscott/sept20_2016.pdf`

7.2. Compañeros con quienes colaboré

- Jose David Gutierrez Uribe
- Jeffrey Garcia Gallego