Resolución 2° Parcial 1C 2021

1 Complejidad

1.1 Funciones

Los primeros tres items los veo con el límite:
$\lim\limits_{n\to+\infty} \frac{c^n}{n^k} = \lim\limits_{n\to+\infty} (\frac{c^{n/k}}{n})^k$

Voy a ver cuánto da el límite entre paréntesis. Como tanto n como $c^{n/k}$ tienden a $+\infty$ (porque $c > 1$ y n/k tiende a $+\infty$) tengo que:
$\lim\limits_{n\to+\infty} \frac{c^{n/k}}{n} =_{L'h} \lim\limits_{n\to+\infty} \frac{ln(c)}{k} c^{n/k} = +\infty$

Porque k y c son constantes positivas mayores a 1 y en consecuencia ln(c)/k también. Entonces, como lo que está entre paréntesis tiende a $+\infty$ y k es una constante mayor a 1 tengo que:
$\lim\limits_{n\to+\infty} (\frac{c^{n/k}}{n})^k = +\infty$

Entonces, por propiedad del límite llego a que $n^k \in O(c^n)$ y $n^k \notin \Omega(c^n)$, y en consecuencia $n^k \notin \Theta(c^n)$.

Los segundos tres items son todos correctos. Uso la propiedad de logaritmos de cambio de base: $log_k(n) = log_c(n)/log_c(k)$. Además, $log_c(k)$ es una constante porque c y k son constantes, y como $c, k \in \mathbb{Z}_{>1}$ entonces $log_c(k) > 0$. Entonces tomo $a = 1/log_c(k)$ (ya sé que $a \in \mathbb{R}_{>0}$) y $n_0 = 1$ y tengo que:
$\forall n \geq n_0 : a * log_c(n) \leq log_k(n) \leq a * log_c(n)$. Ambas desigualdades valen porque vale la igualdad. Entonces por definición tengo que $log_k(n) \in O(log_c(n))$ y $log_k(n) \in \Omega(log_c(n))$, y en consecuencia $log_k(n) \in \Theta(log_c(n))$

Los otros tres los veo usando la propiedad del límite:
$\lim\limits_{n\to+\infty} \frac{(cn)^k}{(kn)^c} = \lim\limits_{n\to+\infty} \frac{c^k n^k}{k^c n^c} = \lim\limits_{n\to+\infty} \frac{c^k}{k^c} \frac{n^k}{n^c} = \lim\limits_{n\to+\infty} \frac{c^k}{k^c} n^{k-c}$.
Como c y k son constantes positivas, entonces $\frac{c^k}{k^c}$ también es una constante positiva (no necesariamente entera, pero seguro una constante positiva). Además, como $k, c \in \mathbb{Z}_{>1}$ y $k > c \implies k - c \geq 1$. Por lo tanto:
$\lim\limits_{n\to+\infty} \frac{(cn)^k}{(kn)^c} = \lim\limits_{n\to+\infty} \frac{c^k}{k^c} n^{k-c} = +\infty$.

Por propiedad del límite, llegamos a que, si $k > c \implies (cn)^k \in \Omega((kn)^c)$ y $(cn)^k \notin O((kn)^c)$, y entonces $(cn)^k \notin \Theta((kn)^c)$

De los últimos tres, el primero lo veo por inducción:
Quiero ver que si $k > c$ entonces $c^n \in O(k^n)$, es decir, $\exists b \in \mathbb{R}_{>0}$ constante y $n_0 \in \mathbb{N}$ tal que $\forall n \geq n_0 : c^n \leq bk^n$.
Caso base: n=1. $c^n = c < k$. Entonces tomando b=1 y $n_0 = 1$ vale la

desigualdad.
Paso inductivo: Supongo que para $n_0 = 1, b = 1 : \; c^{n'} \leq k^{n'} \; \forall n' < n$. Quiero ver que la propiedad vale también para n. Veo que:

$$c^n = c^{n-1} * c \leq_{HI} k^{n-1} * k = k^n$$

Entonces concluyo que tomando $b = 1, n_0 = 1$ entonces $\forall n \geq n_0 : \; c^n \leq bk^n$. Por lo tanto, por definición, $c^n \in O(k^n)$.

Ahora quiero ver que $c^n \notin \Omega(k^n)$. Puedo verlo con un contraejemplo. Tomo c=2 y k=3, que cumplen que $k > c$. Viendo el límite:

$$\lim_{n \to +\infty} \frac{2^n}{3^n} = \lim_{n \to +\infty} (2/3)^n = 0$$

porque $2/3 < 1$. Entonces, por propiedad del límite, $2^n \notin \Omega(3^n)$.

Por último, como $c^n \notin \Omega(k^n) \implies c^n \notin \Theta(k^n)$.

1.2 Análisis de Algoritmos

Una matriz se dice en degradé si:

1. Es cuadrada y todos sus elementos son naturales.
2. Todos sus elementos son distintos.
3. Las filas y las columnas están ordenadas de forma creciente.

1. Sea $A \in \mathbb{N}^{nxn}$, el mejor caso se da cuando el valor v que busco es menor al primer elemento de la matriz (y en consecuencia es menor a todos los elementos de la matriz). Entonces si $v < A[0, 0]$ la función ni siquiera entra a los dos primeros ciclos, porque no se cumple la condición de $A[0, 0] \leq v$. Como nunca entra, al salir de ambos ciclos i vale 0, entonces colLim=filLim=i-1=-1. Cuando llegue al tercer ciclo tampoco va a entrar, porque va a quedar para $i = 0... - 1.$, así que nunca va a entrar al ciclo y va a devolver false. Entonces nos queda que el algoritmo realiza únicamente una cantidad constante de asignaciones, accesos a la matriz (sólo al preguntar la primera vez si $A[0, 0] \leq v$ en los dos primeros ciclos), comparaciones, y una obtención del tamaño de la matriz, lo cual es simplemente una cantidad constante de operaciones $\Theta(1)$, entonces la complejidad en mejor caso del algoritmo es $\Theta(1)$.

2. El peor caso se da cuando el valor v que busco es mayor a todos los elementos de la matriz (es decir, $v > A[n-1, n-1]$). En este caso, tanto el primer como el segundo ciclo se van a ejecutar n veces (hasta que i=n), ya que en cada iteración j se va a cumplir que $v > A[0, j]$ para el primer ciclo y $v > A[j, 0]$ para el segundo. Como las únicas operaciones dentro de los ciclos son una suma y una asignación, que toman tiempo constante, y las comparaciones de la guarda

toman tiempo constante, entonces cada ciclo realiza $\sum_{i=0}^{n-1} \Theta(1)$ operaciones y $\sum_{i=0}^{n-1} \Theta(1) = n * \Theta(1) = \Theta(n)$.
Cuando sale de cada uno de los ciclos, debe ser porque se cumple que i=n, entonces colLim=filLim=i-1=n-1. Por lo tanto, el tercer ciclo se ejecutará también n veces, y en cada iteración de este ciclo, el loop interno se ejecutará también n veces. Dentro del loop interno, todas las operaciones que se realizan llevan tiempo constante, porque son solamente accesos a la matriz y comparaciones de naturales, y nunca va a entrar al if porque $val \notin A$. Entonces, la cantidad de operaciones de estos ciclos anidados es:

$$\sum_{i=0}^{n-1}\sum_{j=0}^{n-1} \Theta(1) = \sum_{i=0}^{n-1} n * \Theta(1) = \sum_{i=0}^{n-1} \Theta(n) = n * \Theta(n) = \Theta(n^2)$$

Entonces, juntando todo la complejidad del algoritmo en el peor caso queda:

$$\Theta(1)+\Theta(n)+\Theta(n)+\Theta(n^2) = 2\Theta(n)+\Theta(n^2) = \Theta(n)+\Theta(n^2) = \Theta(max(n, n^2)) = \Theta(n^2)$$