Pauta de Corrección

Rezago Segundo Certamen
Algoritmos y Complejidad

8 de mayo de 2017

1. Esto es similar al problema de ordenar archivos. La misma idea sirve (las probabilidades son todas 1), y la misma demostración.

   Sea t_i el tiempo que se demora la tarea i, ordenados tal que $t_1 \leq t_2 \leq \cdots \leq t_n$. Ejecutar las tareas en este orden minimiza la suma:

   $$T = \sum_{1 \leq i \leq n} \sum_{1 \leq k \leq i} t_i$$

   Esto lo demostramos por contradicción. Llamemos T^* a la suma en el orden óptimo, que no es en orden creciente de duración. Supongamos que las tareas se ejecutan en un orden distinto, que da el valor mínimo $T^\dagger$. De ser así, hay tareas adyacentes t_r y t_{r+1} tales que $t_r > t_{r+1}$. Si las intercambiamos, solo los instantes de término de estas dos se ven afectados. Evaluamos el cambio:

   $$\begin{aligned} T^* - T^\dagger &= (t_r + (t_r + t_{r+1})) - (t_{r+1} + (t_{r+1} + t_r)) \\ &= t_r - t_{r+1} \\ &> 0 \end{aligned}$$

   O sea, para el supuesto orden óptimo la suma T^* es mayor que la suma $T^\dagger$ para un orden distinto, contradiciendo que el orden era óptimo.

   Otra manera de verlo es que hemos demostrado que si nos dan las tareas en cualquier orden, podemos mejorar el valor de la suma intercambiando tareas adyacentes fuera de orden de duración creciente. Básicamente es usar el método de la burbuja para ordenarlos, mientras obtenemos menores valores para la suma con cada intercambio.

Puntajes

Total	25
Por contradicción	6
Tareas adyacentes “fuera de orden”	7
Cambio en la suma	7
Conclusión	5

2. Analizamos cada algoritmo por turno. Usaremos $A(n)$ al tiempo de ejecución de A para tamaño n, y similarmente para los demás.

   Algoritmo A: La recurrencia es:

   $$A(n) = 3A(n/2) + \Theta(\sqrt{n})$$

   En el teorema maestro, tenemos $a = 3$, $b = 2$, valor crítico es $\log_3 2 > 1/2$ (es $\log_3 \sqrt{3} = 1/2$, y $2 > \sqrt{3}$; en realidad $\log_3 2 = 0,63093$). Por el torpedo, como $f(n) = O(n^{1/2})$, estamos en el primer caso del teorema maestro:

   $$A(n) = O(n^{\log_3 2})$$

   y es claro que $\log_3 2 < 1$.

   Algoritmo B: La recurrencia es:

   $$B(n) = 7B(n/7) + \Theta(n)$$

   Tenemos $a = b = 7$, el valor crítico es $\log_7 7 = 1$. Tenemos $f(n) = \Theta(n \log^0 n)$, vale decir $\alpha = 0 > -1$ es el cuarto caso:

   $$B(n) = \Theta(n \log n)$$

   Algoritmo C: Acá es:

   $$C(n) = 2C(n/2) + \Theta(n \log n)$$

   con $a = b = 2$, valor crítico es $\log_2 2 = 1$; nuevamente el cuarto caso, ahora con $\alpha = 1$:

   $$C(n) = \Theta(n \log^2 n)$$

   Algoritmo D: Acá es diferente. La recurrencia (para un valor conveniente de f) es:

   $$D(n) = 2D(n-1) + f$$

   Dicho de otra forma:

   $$D(n+1) > 2D(n)$$

   de donde se ve claramente que:

   $$D(n) \geq D(0) \cdot 2^n$$

   En realidad, podemos resolver esto. Definamos:

   $$d(z) = \sum_{n \geq 0} D(n) z^n$$

Aplicando propiedades a la recurrencia escrita como:

$$D(n+1) = D(n) + f$$

tenemos:

$$\frac{d(z) - D(0)}{z} = 2d(z) + \frac{f}{1-z}$$

De acá despejamos:

$$\begin{aligned} d(z) &= \frac{(f - D(0))z + D(0)}{1 - 3z + 2z^2} \\ &= \frac{f + D(0)}{1 - 2z} + \frac{f}{1-z} \end{aligned}$$

Obtenemos:

$$\begin{aligned} D(n) &= (D(0) + f) \cdot 2^n + f \\ &= \Theta(2^n) \end{aligned}$$

La mejor opción es A, es la que asintóticamente crece más lento.

Puntajes

Total	35
Análisis de A	8
Análisis de B	8
Análisis de C	8
Análisis de D	8
Conclusiones	3

3. Vemos que interesa el largo del intervalo $[i, j-1]$ en el tiempo de ejecución. Sea $T(k)$ el tiempo de ejecución de SlowHeap(i, j) cuando $j - i = k$. Del algoritmo, la recurrencia exacta es:

   $$T(n) = T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil - 1) + O(n)$$

   Podemos aplicar el teorema maestro a la recurrencia aproximada:

   $$T(n) = 2T(n/2) + O(n)$$

   Esto es $a = 2, b = 2, f(n) = O(n)$, el valor crítico es $\log_2 2 = 1$. Se aplica el cuarto caso del teorema maestro, con $\alpha = 0$:

   $$\begin{aligned} T(n) &= \Theta(n^{\log_2 2} \log^{0+1} n) \\ &= \Theta(n \log n) \end{aligned}$$

Puntajes

Total	30
Plantear la recursión	10
Aplicar el teorema maestro	20

4. Si consideramos las denominaciones $d_0, \ldots, d_m$ y la cantidad n, las formas de dar cambio se dividen en dos grupos:

   Las que incluyen d_m: Es una moneda más que la mejor manera de dar cambio de $n - d_m$ con las denominaciones $d_0, \ldots, d_m$

   Las que no incluyen d_m: Es la mejor manera de dar cambio de n con las denominaciones $d_0, \ldots, d_{m-1}$

   Esto da la recurrencia para $c[m, n]$, el número mínimo de monedas de $d_0, \ldots, d_m$ para dar cambio de n:

   $$c[m, n] = \min\{1 + c[m, n - d_m], c[m - 1, n]\} \qquad c[m, 0] = 0, c[0, n] = n$$

   Las condiciones de contorno son que si no hay nada que dar, no damos monedas; si solo tenemos disponibles monedas $d_0 = 1$, lo único que podemos hacer es entregar n monedas de uno.

   Podemos registrar lo anterior en un arreglo, que llenamos para $m = 0, 1, \ldots$ para los distintos $n = 0, 1, \ldots$

Puntajes

Total	30
Plantear la recurrencia	15
Explicar arreglo y orden de llenado	15

HvB/LaTeX2ε