Ayudantía 12 - The last one

Sassy complexes

1. Problemas

1. Para cierto problema de tamaño n cuenta con tres algoritmos alternativos:

   Algoritmo A: Ordena los datos dados, divide en tres problemas de un tercio de tamaño y combina en tiempo constante.

   Algoritmo B: Resuelve recursivamente dos problemas de tamaño $n-1$ y combina las soluciones en tiempo constante.

   Algoritmo C: Divide el problema de tamaño n en nueve de tamaño $n/4$ y combina las soluciones en tiempo $O(n^2)$.

   ¿Cuál elije si n es grande, y porqué?

2. Se dan n puntos x_i en la recta, que deben cubrirse con el mínimo número de intervalos unitarios $[s_i, s_i+1]$ (el intervalo cubre todos los x que caen dentro del intervalo). Plantee un algoritmo eficiente para resolver este problema.

3. Para cierta aplicación requiere mantener una lista de elementos, sobre la que se efectúan dos operaciones:

   insert(k): Inserta un nuevo elemento de valor k a la lista. Tiene costo 1.

   sum(): Suma los elementos de la lista, los elimina y los reemplaza por un elemento que es la suma. El costo es n si la lista tiene n elementos.

   Demuestre que el costo amortizado de insert(k) es $O(1)$, y que el costo amortizado de sum() es $O(1)$.

4. Dadas denominaciones de monedas $d_0 = 1 < d_1 < \cdots < d_k$, el problema de cambio es hallar el mínimo número de monedas para entregar la cantidad n. Plantee un algoritmo eficiente para obtener el mínimo número de monedas requerido. Note que el algoritmo voraz de entregar lo que se pueda de la máxima denominación no siempre funciona, por ejemplo con denominaciones $1, 5, 20, 25$ para 40 da $25, 5, 5, 5$ (4 monedas), el óptimo es $20, 20$ (2 monedas).

5. Dada una moneda sesgada que al lanzarla sale cara con probabilidad p y sello con probabilidad $(1-p)$, proponga una forma de simular el lanzamiento de una moneda sin sesgos ($p = 0{,}5$) con esta moneda. Demuestre. *Hint: Para demostrar utilice el teorema de Bayes*

2. Soluciones

1. Llamemos $A(n)$, $B(n)$ y $C(n)$ los tiempos de ejecución respectivos. Del problema extraemos las recurrencias.

   Algoritmo A: Ordenar n elementos tiene costo $\Theta(n \log n)$. Recurrencia es $A(n) = 3A(n/3) + \Theta(n \log n)$. Es aplicable el teorema maestro, $a = b = 3$, $f(n) = \Theta(n \log n)$. El valor crítico es $\log_3 3 = 1$, tenemos que $f(n) = \Theta(n \log n)$, es el caso 4 del teorema maestro con $\alpha = 1$, resulta:

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

   Algoritmo B: Recurrencia es $B(n) = 2B(n-1) + \Theta(1)$. Definamos la función generatriz:

   $$b(z) = \sum_{n \geq 0} B(n) z^n$$

   Aplicando las propiedades a la recurrencia, aproximando $f(n) = c$ queda:

   $$\frac{b(z) - B(0)}{z} = 2b(z) + \frac{c}{1-z}$$

Despejando:

$$b(z) = \frac{B(0)}{1-2z} + \frac{c}{(1-z)(1-2z)}$$

Por fracciones parciales sabemos que esto puede escribirse:

$$b(z) = \frac{\alpha}{1-2z} + \frac{\beta}{1-z}$$

Sabemos que de acá:

$$\begin{aligned} B(n) &= [z^n]b(z) \\ &= \alpha \cdot 2^n + \beta \\ &= \Theta(2^n) \end{aligned}$$

Algoritmo C: La recurrencia es $C(n) = 9C(n/4) + \Theta(n^2)$, Para el teorema maestro, $a = 9$, $b = 4$, $f(n) = \Theta(n^2)$. El valor crítico es $\log_4 9 < 2$, estamos en el caso 5 si se cumple la condición adicional con $c = 2$. Verificamos ésta:

$$9 \cdot \left(\frac{n}{4}\right)^2 = \frac{9}{16}n^2$$

O sea, se cumple por ejemplo con $k = 5/8 < 1$. Concluimos que:

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

Para n grande, el mejor es el algoritmo A.

2. Un algoritmo voraz obvio es ordenar los puntos en orden de x creciente, y luego iniciar el primer intervalo en el primer x, cubriendo todos los que están en $[x, x+1]$, y repetir con los no cubiertos.

   Veamos nuestras tres propiedades, suponiendo que los puntos están ordenados tal que $x_1 < x_2 < \cdots < x_n$.

   Elección voraz: Toda solución debe cubrir x_1, no tiene sentido que el intervalo que lo cubre no comience en x_1 (¿Para qué cubrir puntos anteriores a x_1?). O sea, hay una solución óptima que incluye el intervalo $[x_1, x_1 + 1]$.

   Estructura inductiva: Elegir el intervalo $[x_1, x_1 + 1]$ cubre algunos x, que quedan excluidos; debemos cubrir a los restantes, y el intervalo elegido no interfiere en ello. No hay restricciones externas.

   Subestructura óptima: La solución óptima de cubrir a los que no caen en $[x_1, x_1 + 1]$ se combina con ese intervalo para dar solución óptima al problema.

   Vimos que si se dan las tres propiedades, el algoritmo voraz da una solución óptima.

3. Para análisis amortizado, la alternativa que se ve más simple es el método potencial. Definimos la función potencial, donde n es el largo actual de la lista:

   $$\Phi(n) = n$$

   Es claro que $\Phi(n) \geq 0$, y el potencial inicial es $\Phi(0) = 0$.

   Consideremos la secuencia de operaciones σ_i, de costos respectivos c_i y costos amortizados a_i, con estados s_i luego de la i-ésima operación:

   $$a_i = c_i + \Phi(s_i) - \Phi(s_{i-1})$$

   Si σ_i es insert(k), y en ese momento el largo de la lista era n, es:

   $$\begin{aligned} a_i &= 1 + \Phi(n+1) - \Phi(n) \\ &= 1 + n + 1 - n \\ &= 2 \end{aligned}$$

   Si σ_i es add(), y en ese momento el largo de la lista era n, es:

   $$\begin{aligned} a_i &= n + \Phi(1) - \Phi(n) \\ &= n + 1 - n \\ &= 1 \end{aligned}$$

   O sea, el costo amortizado de insert(k) es 2, el de add() es 1, ambos $O(1)$ como se indica.

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:

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

   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 $n = 0, 1, \ldots$

5. WIP