CC40A - Control 1

Prof. Gonzalo Navarro

27 de Mayo de 2009

El examen, como puede ver, suma 8.5 puntos. Los excesos sobre 7.0 se premiarán con una felicitación, pero no son reciclables :-). Se recomienda resolver primero las preguntas marcadas con ($\star$), que son más fáciles y/o rápidas, y suman más que suficiente como para aprobar si están todas correctas. Deje los bonos para el final; le llevarán más tiempo y el costo/beneficio es probablemente inferior al de resolver las preguntas principales.

P1 (2.5 pt)

Dado un arreglo desordenado de elementos que sólo pueden compararse por $<$, $=$, $>$, se desean obtener los k menores elementos del arreglo, en orden creciente.

1. ($\star$) (0.5pt) Diseñe un algoritmo que lo resuelva en tiempo $O(n + k \log k)$.
2. (1pt) Demuestre que este algoritmo es óptimo.
3. ($\star$) (1pt) Considere ahora la variante *online* del problema, donde no se conoce k sino que el usuario va pidiendo el siguiente mínimo, hasta que se detiene luego de haber pedido k de ellos, sin que sea posible saber cuándo se detendrá. Diseñe un algoritmo óptimo para este problema. (Para ello, puede serle útil saber que $n + k \log n = \Theta(n + k \log k)$; hay un bono de 0.5 pt para quienes demuestren esto.)

P2 (2.5 pt)

Se tienen n símbolos con probabilidades $p_1, \ldots, p_n$. Se desea construir un código prefijo alfabético óptimo para estas probabilidades. Recuerde que este código equivale a un árbol donde las hojas tienen etiquetas ordenadas $1, \ldots, n$, y se minimiza $L = \sum_{i=1}^{n} p_i \ell_i$, donde ℓ_i es la profundidad de la i-ésima hoja. En el curso se resolvió este problema para códigos binarios usando programación dinámica (considere por simplicidad el algoritmo de tiempo $O(n^3)$, y el problema de únicamente determinar el mejor L).

1. ($\star$) (1pt) Suponga que, por razones físicas, transmitir un 1 tiene un costo distinto del de transmitir un cero, $c(1) \neq c(0)$. Entonces quisiéramos redefinir ℓ_i como la suma de los $c(\cdot)$ de las aristas del árbol que se recorren para llegar de la raíz a la i-ésima hoja. Modifique el algoritmo visto en clase para resolver esta variante.
2. (1.5pt) Suponga nuevamente que los costos son todos iguales, pero ahora el alfabeto de codificación no es binario, sino que tiene k símbolos, por lo cual el árbol es k-ario. Modifique el algoritmo visto en clase para resolver esta variante, y analícelo.

Nota: suponga que n es tal que el código puede componerse con árboles exactamente k-arios (técnicamente esto es $n \equiv 1 \bmod (k-1)$), por lo tanto puede considerar sólo subproblemas cuyo tamaño cumpla esa condición.

Nota': es de esperar que su costo resulte ser $O(n^{k+1})$. Hay un bono de 1pt para quienes logren reducir esto a $O(kn^3)$ (se puede conseguir también con programación dinámica).

P3 (2.5 pt)

Un *árbol α-balanceado*, para $1/2 < \alpha < 1$, es un árbol binario de búsqueda donde todo subárbol $T = (root, T_l, T_r)$ cumple $|T_l| \leq \alpha \cdot |T|$ y $|T_r| \leq \alpha \cdot |T|$. Las operaciones para buscar y mantener un árbol α-balanceado son las mismas que para un árbol binario de búsqueda, excepto que luego de insertar o borrar un nodo, se busca el nodo más alto en el camino del punto de inserción/borrado hacia la raíz, que no esté α-balanceado, y se lo reconstruye como árbol perfectamente balanceado (el costo es proporcional al tamaño del subárbol que se reconstruye).

1. ($\star$) (1pt) Muestre que la búsqueda en un árbol α-balanceado cuesta $O(\log n)$, y que lo mismo ocurre con las inserciones y borrados, si no consideramos las reconstrucciones. ¿Qué constante obtiene multiplicando el $\log n$?

2. (1.5pt) Muestre que el costo amortizado de las inserciones y borrados, ahora considerando reconstrucciones, es también $O(\log n)$. Para ello, considere la función potencial

$$\Phi(T) \;=\; \frac{1}{2\alpha - 1} \sum_{T' \in T} abs(|T'_l| - |T'_r| - 1),$$

donde $abs(\cdot)$ es el valor absoluto, y $T' \in T$ significa que T' es un subárbol de T. Hay un bono de 0.5pt por encontrar la constante que multiplica a este $\log n$, y recomendar un α óptimo.

Tiempo: 2.5 horas | Con una hoja de apuntes