CC4102 - Examen

Prof. Gonzalo Navarro

20 de Diciembre de 2021

P1 (2.0 pt)

Se desea una estructura de datos para almacenar un conjunto de strings en memoria secundaria. Estos strings pueden ser bastante largos, posiblemente ocupando varias páginas de disco.

1. Diseñe una variante directa del B-tree para almacenar estos strings: describa la estructura de datos y cómo se busca (no necesita considerar los algoritmos de inserción y borrado). Analice el costo de búsqueda en el modelo de memoria secundaria e indique qué problema tiene su estructura comparada con un B-tree de números.

2. Considere ahora una variante donde en cada nodo del B-tree se guarda un árbol Patricia de las claves en vez de las claves directamente. Nuevamente describa la estructura y el algoritmo de búsqueda, y analice el costo de búsqueda. ¿Cómo se compara el costo con el de un B-tree de números? ¿Es el resultado óptimo en el modelo de comparaciones?

P2 (2.0 pt)

Se tiene una secuencia $P[1,n]$ de números reales positivos y un rango real positivo $[x,y]$, $0 < x \leq y$. Se desea encontrar todas las posiciones j tal que, para algún $i \leq j$, $x \leq \sum_{k=i}^{j} P[k] \leq y$.

1. Demuestre que el problema no se puede resolver con menos de n accesos a P.

2. Diseñe un algoritmo de tiempo $O(n)$ que resuelva el problema. Demuestre su correctitud y use alguna técnica de análisis amortizado para analizarlo.

P3 (2.0 pt)

Dado un grafo no dirigido $G = (V,E)$, considere el problema de dividir $V = S \cup S'$, con $S' = V - S$, de manera de maximizar la cantidad de aristas que cruzan entre S a S'.

1. Considere el algoritmo que toma cada vértice y tira una moneda para decidir si lo pone en S o S'. Pruebe que, en el caso esperado, se obtiene una 2-aproximación. Hint: considere que, si de v_i salen e_i aristas hacia $v_1, \ldots, v_{i-1}$, al poner v_i en S habrá k_i aristas que cruzan a S' y al ponerlo en S' habrá $e_i - k_i$ aristas que cruzan a S.

2. Considere el algoritmo determinístico que toma cada vértice v_i, para $i = 1, \ldots, |V|$, y lo pone en S o en S' según en cuál de los dos se maximice la cantidad de aristas entre v_i y $v_1, \ldots, v_{i-1}$ que cruzan entre S y S'. Pruebe que este algoritmo es una 2-aproximación.

Tiempo: 3 horas