CC30A Algoritmos y Estructuras de Datos – Control 2

Profesor: Patricio Poblete

Junio 9, 2000

1. Se define que un árbol de búsqueda binaria es de *peso balanceado* si, para todo nodo interno con subárboles I y D, se tiene que sus respectivos números de nodos externos N_I y N_D satisfacen

$$\alpha \leq \frac{N_I}{N_I + N_D} \leq 1 - \alpha,$$

donde α es un parámetro ($0 \leq \alpha \leq 1/3$).

   a) Encuentre la altura que un árbol de este tipo puede alcanzar en el peor caso, en función de su número N de nodos externos. (Indicación: Considere la familia de los árboles más desbalanceados posibles y encuentre una ecuación de recurrencia para la altura.)

   b) Para implementar estos árboles, es conveniente llevar almacenado en cada nodo interno el peso del subárbol que lo tiene como raíz (peso = número de nodos externos de ese subárbol). Muestre cómo actualizar estos pesos almacenados para el caso de los nodos internos involucrados en una rotación simple y en una rotación doble.

2. *a*) Defina una clase abstracta `Nodo`, y clases derivadas de ella `Nodo2`, `Nodo3` y `Hoja`, para implementar árboles 2-3. Para simplificar, suponga que los campos `info` son de tipo `int`.

   b) Implemente en estas clases un método `imprimir`, de modo que si `a` es de tipo `Nodo`, entonces la llamada `a.imprimir()` recorra el árbol e imprima todas las llaves en orden ascendente.

   c) Implemente además un método `buscar` tal que `a.buscar(x)` entregue verdadero o falso, dependiendo si la llave `x` está en el árbol o no.

3. Los dos problemas siguientes suponen que los elementos tienen probabilidades de acceso diferentes y conocidas de antemano.

   a) Demuestre que el orden óptimo de los elementos para una búsqueda secuencial es en orden decreciente de probabilidad.

   b) Cosntruya un ejemplo de un árbol con tres elementos que muestre que no siempre es óptimo poner como raíz del árbol al elemento más probable.

Tiempo: 2 horas Entregar en hojas separadas Con apuntes de clases