Estructura de datos
Árboles B
Actividad 8

Dagoberto Quevedo

18 de marzo de 2020

Resumen

En esta actividad se describen las propiedades y funcionalidades de los árboles B, complementando con el uso de un librería en python que facilita su implementación computacional.

1. Árboles B

Los árboles B son una generalización de los árboles binarios, que resulta eficiente en el modelo de memoria externa. Para cualquier entero $B \geq 2$, un árbol B es un árbol en el que todas las hojas tienen la misma profundidad y cada nodo no raíz, u, tiene al menos B hijos y como máximo $2B$ hijos. Los hijos de u se almacenan la lista, c_u. El número de hijos permitidos por nodo es entre 2 y $2B$ hijos. En los árboles B aplica para la altura ℓ del árbol que $B \geq 2$,

$$h \leq \log_B \frac{n+1}{2}, \tag{1}$$

Si la altura de un árbol B es h, entonces se deduce que el número, ℓ, de hojas en el árbol B es,

$$2B^{h-1} \leq \ell \leq 2(2B)^{h-1}. \tag{2}$$

Es decir, la altura de un árbol B es proporcional al logaritmo base B del número de hojas. Cada nodo u, en B almacena una lista de claves $c_i^u, \ldots, c_{2B-1}^u$. Si u es un nodo interno con k hijos, entonces el número de claves almacenadas en u es exactamente $k-1$ y se almacenan en $c_1^u, \ldots, c_{k-2}^u$. Las claves en un árbol B se respetan de manera similar que en un árbol de búsqueda binario. Dado un nodo, u que almacena las claves $k-1$,

$$c_1^u < c_2^u, \ldots, c_{k-2}^u \tag{3}$$

Búsqueda de una clave en un árbol B es distinto a una operación de búsqueda de un árbol binario, sólo que en esta clase de árboles se tendrá que elegir entre distintas alternativas en cada nodo intermedio. La siguiente figura muestra un ejemplo de árbol B,

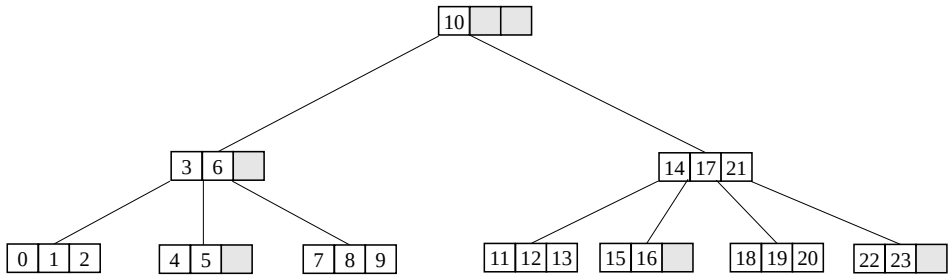

Figura 1: Ejemplo de un árbol B, con $B = 2$.

Tenga en cuenta que los datos almacenados en un nodo de un árbol B tienen un tamaño B. Por ejemplo, si las claves son números enteros de 4 bytes y los índices del nodo también son 4 bytes, entonces establecer $B = 256$ significa que cada nodo almacena $(4 + 4) \times 2B = 8 \times 512 = 4096$ bytes.

1.1. Implementación computacional

Se hace uno de la librería `BTree` para la implementación computacional; el diseño experimental consiste en la la creación aleatoria de árboles B, dado un cierto dominio de número enteros y medir su eficiencia en la búsqueda de casos extremos, así como el tamaño de las estructuras en memoria.

Referencias

[1] *Open Data Structures*, 14.2 B-Trees, `opendatastructures.org/ods-python/14_2_B_Trees.html`

[2] Donald Knuth, *Sorting and searching*, The Art of Computer Programming, Addison-Wesley Professional, 1998.

[3] Elisa Schaeffer, *Modelos computacionales*, Complejidad computacional de problemas y el análisis y diseño de algoritmos, notas de curso, 2020.