Propiedades de árboles.
Facultad de Ciencias
Estructuras de Datos

Verónica E. Arriola Ríos
v.arriola@ciencias.unam.mx

Luis Fernando Yang Fong Baeza
fernandofong@ciencias.unam.mx

José Antonio Vilchis Salazar
grand_paladin@ciencias.unam.mx

4 de Mayo 2020

1. Introducción

Este pequeño PDF relata sobre propiedades que son siempre válidas sobre árboles binarios bien construidos (balanceados) como lo son los Árboles Binarios Completos o algún árbol binario autobalanceable, como sus alturas o el número de elementos que tiene cada un árbol o distintas propiedades, léease con detenimiento y cuidado tratando de entender las ideas detrás de cada demostración, no solo leerla, sino también saber que en cualquier algoritmo que utilice una estructura de árbol binario, se pueden aplicar estas propiedades.

Se sugiere también tener en claro conceptos como inducción e inducción fuerte puesto que por la naturaleza de los árboles binarios, probar sus propiedades bajo inducción resulta mucho más fácil.

2. Definiciones básicas

1. Árbol binario := Es un TDA que almacena un vértice, que contiene un elemento, llamado vértice raíz y tiene un subárbol binario izquierdo y un subárbol binario derecho, un árbol binario puede ser vacío.
2. Hoja := Un vértice de un árbol, cuyos subárboles izquierdo y derecho son vacíos.
3. Nodo interno := Vértice de un árbol binario cuyos subárboles izquierdo o derecho no son vacíos.
4. Árbol binario de búsqueda := Un árbol binario que cumple con la propiedad de que cualquier elemento en el subárbol izquierdo es menor o igual al elemento almacenado en la raíz y cualquier elemento en el subárbol izquierdo es mayor al elemento almacenado en la raíz.
5. Árbol binario completo := Árbol binario cuyos vértices cumplen con alguna de las siguientes tres propiedades:
   a) Es hoja.
   b) Sus subárboles izquierdo y derecho no son vacíos.
   c) Si alguno de sus subárboles es vacío, ese subárbol tiene que ser una hoja.
6. Altura := El camino más largo desde el vértice raíz hasta una hoja del árbol, se cuenta como camino, por el número de nodos internos que se recorren hasta la esa hoja.

3. Propiedades

Lema El número de nodos internos de un árbol binario de n hojas, es $n-1$ nodos internos.

Demostración. *Por inducción sobre n*

Caso base. Supongamos que $n=1$, entonces tenemos un árbol binario con una hoja, es decir el árbol más pequeño que podemos construir, $n-1=1-1=0$, es decir, no hay nodos internos, lo cual, es cierto.

Hipótesis de inducción fuerte. Supongamos que para cualquier árbol binario de k elementos, con $k<n$, cumple la hipótesis de inducción.

Paso inductivo. Por demostrar que para un árbol binario con n hojas, tenemos $n-1$ nodos internos. Observemos que para n elementos, podemos crear $\lceil \frac{n}{2} \rceil$ árboles, para que tener las n hojas como se muestra en la figura, pero con esos $\lceil \frac{n}{2} \rceil$ podemos crear un nuevo árbol binario que tiene $\lceil \frac{n}{2} \rceil$ hojas y por hipótesis de inducción, tiene $\lceil \frac{n}{2} \rceil - 1$ nodos internos, haciendo la suma total de nodos internos, vamos a tener que para n elementos, vamos a tener $\lceil \frac{n}{2} \rceil + \lceil \frac{n}{2} \rceil - 1 = n-1$ □

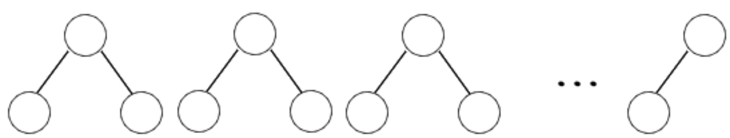

Figura 1: El caso en que n es impar

Lema Sea i la altura de un árbol binario a, entonces a tiene a lo más $2^i - 1$ elementos.

Demostracion *Por inducción sobre n*

Caso base.
Supongamos que $i=0$, entonces tenemos a lo más $2^0-1=0$ elementos, entonces el árbol vacío. Un caso base más concreto, supongamos que $i=1$, entonces tenemos a lo más $2^1-1=1$ elementos, esto es un árbol binario con subárboles vacíos, entonces cumplen la hipótesis de inducción.

Hipótesis de inducción.
Supongamos que para un árbol de altura i, tenemos a lo más 2^i-1 elementos.

Paso inductivo. Todos los nodos del árbol de hipótesis de inducción, se vuelven nodos internos de manera que entonces el número de hojas necesariamente tiene que ser 2^i, más el número de nodos internos que teníamos que es 2^i-1, el número máximo de nodos que podemos tener es la suma de éstos, que haciendo la cuenta es $2^i + 2^i - 1 = 2^(i+1) - 1$ como se esperaba, como puede ser que alguna de las ramas no tengan sus hijos completos, necesariamente tiene que estar acotado por el número máximo, no puede exceder, terminando la demostración. □

Lema La altura de un árbol binario bien balanceado de n es del orden de $O(log\ n)$.

Demostracion Sabemos por el lema anterior que para un número de elementos, podemos encontrar una $i \in \mathbb{N}$, tal que se cumpla lo siguiente:

$$2^{i-1} \leq n \leq 2^i$$

De manera que aplicando las propiedades de desigualdades que dice que si f es una función creciente y $x \leq y \to f(x) \leq f(y)$, entonces aplicando la función log_2, tenemos que:

$$log_2(2^{i-1}) \leq log_2(n) \leq log_2(2^i)$$

Pero trivialmente $log_2(2^n) = n$, simplificando la ecuación de arriba, tenemos que

$$i - 1 \leq log_2(n) \leq i$$

Acotando la altura del árbol, exactamente entre $[i-1, i]$ que es un número en términos de $log_2(n) \in O(log\ n)$ concluyendo la demostración, si $n \leq 2^0$, para ser congruente, entonces hay que acotar el rango como $[i, i+1]$ □

Lema El grado de cada vértice es a lo más 3.

Demostración. Recordando que el grado de un vértice es el número de aristas que inciden en él de la gráfica y que un ejemplo de una gráfica es un árbol, entonces cada vértice trivialmente puede tener a lo más 3, el padre y sus dos hijos, completando la demostración. *Corolario* Si es una gráfica dirigida, entonces tiene grado de salida 2 y grado de entrada 1.

Lema Un árbol tiene un número lineal de aristas.

Demostración. Si tenemos n hojas en el árbol, tenemos $n-1$ nodos internos, de manera que tenemos $2n-1$ vértices o elementos en el árbol, por el lema anterior, cada uno puede tener a lo más 3, sin embargo, esto no es del todo cierto, porque las hojas tienen exactamente grado 2 y la raíz del árbol también, entonces tendríamos a lo más $n+1$ vértices de grado 2 y $n-2$ vértices de grado 3, sumando las aristas, tenemos $2n+1+3n-2 = 5n-1$ aristas totales, a lo más y es trivial demostrar que $5n-1 \in O(n)$, teniendo un número lineal de aristas. *Corolario* Los recorridos sobre árboles, requieren tiempo $O(n)$, pensando en BFS y DFS que tomaban tiempo $O(|V|+|E|)$ pero como $|E| \in O(n)$, entonces $O(|V|) = O(n)$. □

Lema El tiempo de construcción de un árbol binario de búsqueda toma tiempo $O(n\ log\ n)$, por lo visto en clase, insertar un elemento en un árbol binario tiene una complejidad de $O(logn)$, entonces al crear un árbol es equivalente a insertar n elementos, de manera que el tiempo total de construcción va a estar acotado a $O(n\ log\ n)$. *Corolario* Ocupa espacio $O(n)$, puesto que hay que almacenar en memoria los n elementos y las aristas que ya mostramos que es un número lineal de aristas. □