



5. Árboles Binarios Completos

Carlos Zerón Martínez

Universidad Nacional Autónoma de México

zeron@ciencias.unam.mx

Jueves 3 de Diciembre de 2020

Definición

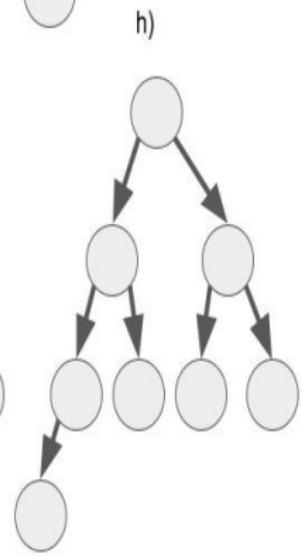
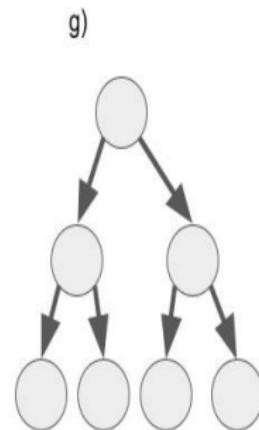
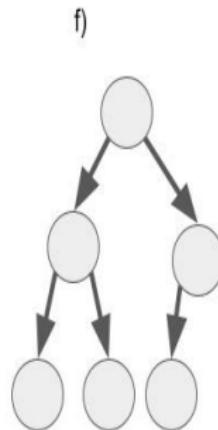
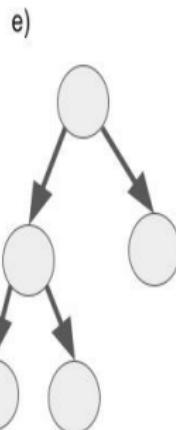
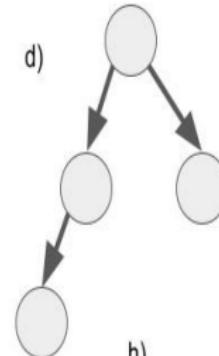
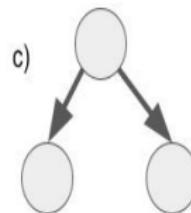
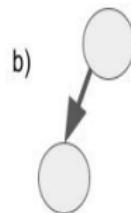
Sea T un árbol binario con profundidad d . Decimos que T es **completo** si cada nivel $i = 0, 1, \dots, d - 1$ tiene 2^i nodos para $0 \leq i \leq d - 1$ y en el nivel d , todos los nodos están colocados lo más a la izquierda posible.

Observaciones:

- ▶ La forma de insertar nodos en un árbol completo es de izquierda a derecha y de arriba hacia abajo, llenando completamente un nivel para comenzar con el siguiente.

- ▶ Un árbol lleno es completo pero un árbol completo no necesariamente es lleno.

Ejemplo



Altura de árboles binarios completos

Teorema *Sea h la altura de un árbol binario completo de tamaño n . Entonces $h = \lfloor \log n \rfloor$*

Sea T un árbol binario completo. Sabemos que h es igual a la profundidad d de T . Vamos a establecer una cota mínima y una máxima para poder obtener el techo.

El número de nodos de T cuando sólo hay un nodo en el último nivel es:

$$n = (2^d - 1) + 1 = 2^d$$

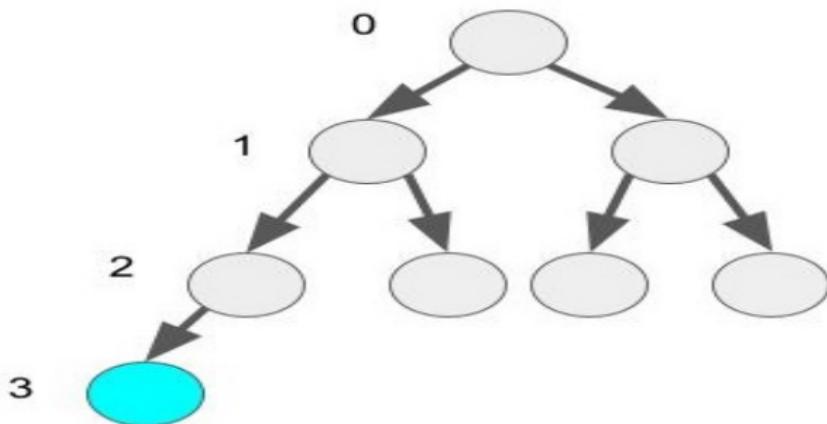
el primer sumando corresponde a los nodos que ocupan los niveles en los niveles anteriores a d . El segundo sumando es el único que ocupa el nivel d de profundidad.

Altura de árboles binarios completos

Teorema Sea h la altura de un árbol binario completo de tamaño n . Entonces $h = \lfloor \log n \rfloor$

El número de nodos de T cuando sólo hay un nodo en el último nivel es:

$$n = (2^d - 1) + 1 = 2^d = 2^h$$



Altura de árboles binarios completos

Teorema *Sea h la altura de un árbol binario completo de tamaño n . Entonces $h = \lfloor \log n \rfloor$*

El máximo de nodos de T es $2^{d+1} - 1 = 2^{h+1} - 1$ (el que tiene un árbol binario lleno con profundidad d)

$$\therefore 2^h \leq n \leq 2^{h+1} - 1 < 2^{h+1}$$

Extrayendo logaritmo de ambos lados, tenemos que:

$$h \leq \log_2 n < h + 1$$

y como h es entero, tenemos que

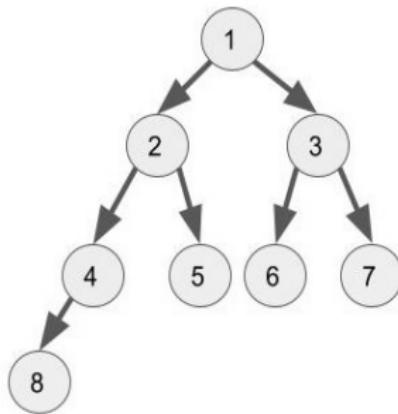
$$\lfloor \log_2 n \rfloor = h$$

Representación de Arboles Binarios Completos

Numeramos los n nodos por niveles y los asignamos a un arreglo de $n + 1$ elementos de la siguiente manera:

- ▶ Se deja vacía la posición 0.
- ▶ Al nodo raíz se le asigna la posición 1 del arreglo.
- ▶ Si un nodo x tiene la posición i , entonces al hijo izquierdo de x se le asigna la posición $2i$ y al hijo derecho, la posición $2i + 1$.
- ▶ Si además x es un nodo que ocupa una posición $i \neq 1$, entonces su nodo padre está en la posición $\lfloor i/2 \rfloor$.

Representación de Arboles Binarios Completos



X								
0	1	2	3	4	5	6	7	8

Ejemplos:

- ▶ el hijo izquierdo del nodo en la posición 3 se encuentra en la posición $2(3) = 6$
- ▶ el hijo izquierdo del nodo en la posición 2 se encuentra en la posición $2(2) + 1 = 5$
- ▶ el padre del nodo en la posición 7 se encuentra en la posición 3

Representación de Arboles Binarios Completos

Con esta implementación, los n nodos tienen índices contiguos en el rango de enteros $[1, n]$. Las modificaciones que se hagan a este tipo de árboles deben mantener la propiedad de que los árboles sean completos, que no falte ningún nodo intermedio.

Se podría representar cualquier árbol binario completo de esta manera. Si no se conoce el número de nodos, se puede emplear un arreglo dinámico.

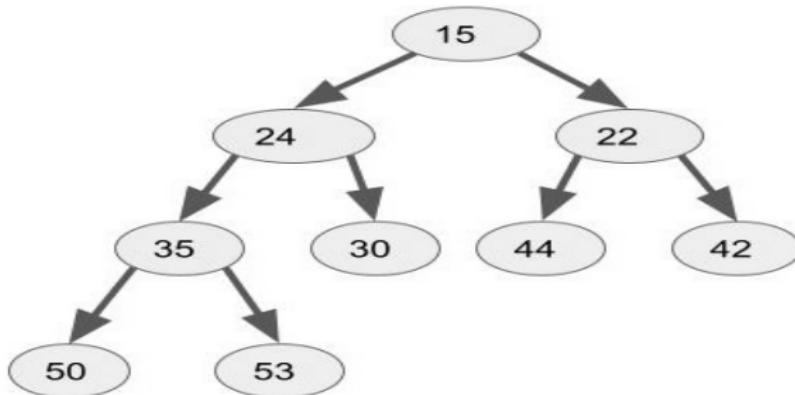
Pueden emplearse para la implementación de colas de prioridades y ordenamiento, donde por lo general ya se conoce el número de nodos.

Heaps mínimos

Un **heap mínimo** es un árbol binario completo que puede ser vacío o bien, satisface las siguientes condiciones:

- ▶ El valor de cada objeto almacenado en los nodos descendientes de la raíz es mayor o igual que el valor del objeto almacenado en la raíz
- ▶ Los subárboles izquierdo y derecho son heaps mínimos.

El heap almacena el mínimo elemento en la raíz

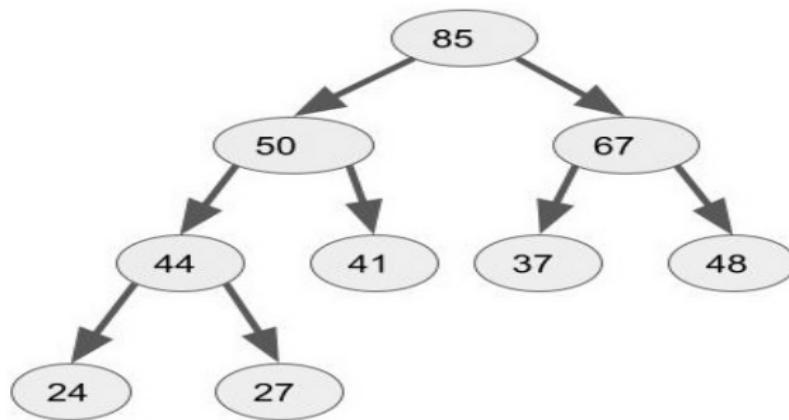


Heaps máximos

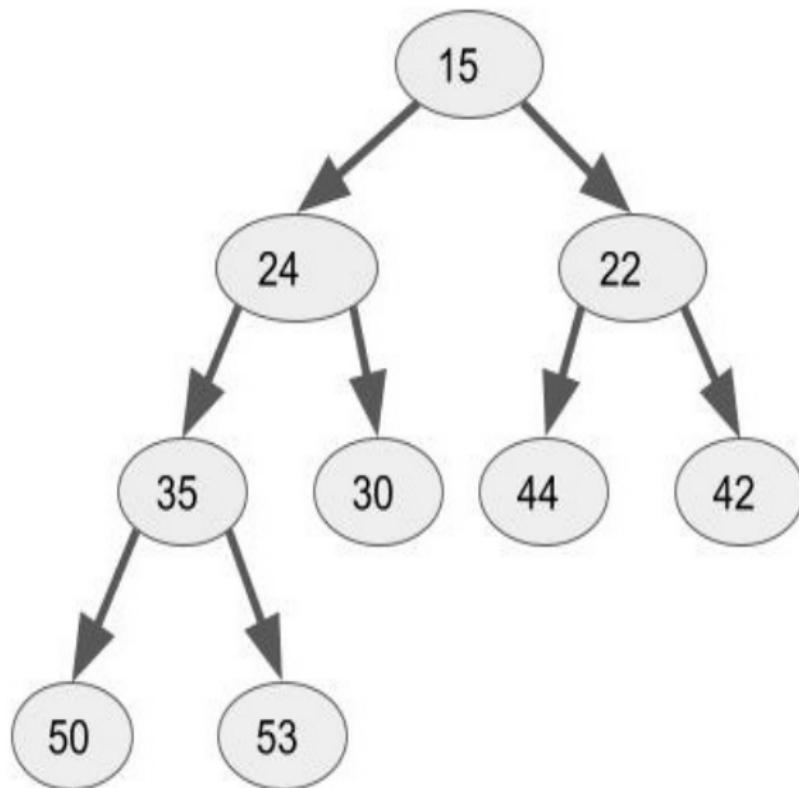
Un **heap máximo** es un árbol binario completo que puede ser vacío o bien, satisface las siguientes condiciones:

- ▶ El valor de cada objeto almacenado en los nodos descendientes de la raíz es menor o igual que el valor del objeto almacenado en la raíz
- ▶ Los subárboles izquierdo y derecho son heaps máximos.

El heap almacena el máximo elemento en la raíz



Heap mínimo en general no es árbol binario de búsqueda



Arbol binario de búsqueda en general no es heap mínimo

