Clase 3 Base de datos

Arboles

Las listas y colas son biunívocas -> para un nodo le entra y le sale una sola flecha

Los arboles son **Univoco** -> a un nodo le entra uno y le sale uno o muchos

Grado de un árbol -> cantidad máxima de hijos o subárboles que puede tener un nodo.

2^grado = cantidad máxima de elementos

Nivel -> Es la posición en la que se encuentra cada nodo relativa a la raíz, es decir, el nivel 0. En una biunívoca, el nivel es igual a la cantidad de elementos. Cuando busco, comparo por niveles, cuando elijo un nodo para buscar, busco en todos los niveles inferiores.

Profundidad -> cantidad de niveles **distintos** que tiene el árbol.

Representación computacional

* Estática -> es un vector. Para cambiar de nivel debo multiplicar mi índice por el grado y para llegar al primer hijo sumo 1y para el segundo sumo 2 y así. Para saber donde estoy, tengo que ver el modulo respecto del grado y restar esa cantidad. Luego divido por el grado y me encuentro en la posición del padre.

Un elemento “i” tiene su hijo en 2i +1 y 2i + 2 y así hasta completar el grado

* Dinámica -> conjunto de nodos de igual tipo con cantidad de nodos igual al grado del mismo.

Características del árbol

* Completo -> todos los nodos del árbol cumplen con el grado o son hijos
* Balanceado -> todos los subárboles de la raíz pesan lo mismo, es decir, tienen la misma cantidad de elementos o una diferencia indivisible (me sobra una cantidad que no puedo dividir por el grado, ya no puedo hacer mas reparticiones).
* Perfectamente balanceado -> un árbol esta balanceado en todos sus niveles.
* Cantidad máxima de elementos en un árbol = (grado ^ niveles) - 1 esto es el crecimiento en los arboles:

A graph with a red line and blue line

AI-generated content may be incorrect.

* Búsqueda en arboles: elementos = grado ^niveles-1 -> elementos + 1 = grado ^ nivel -> niveles = log (elementos + 1) – grado

A graph with a line drawn on it

AI-generated content may be incorrect.

Árbol binario de búsqueda: Es un árbol de grado 2 donde el elemento mas pequeño que la raíz va por la izquierda y el mas grande de la raiz va por derecha

A screenshot of a computer

AI-generated content may be incorrect.

Esto es lo mismo que hacer búsqueda binaria con un vector o lista. Si lo quiero balancear, necesito que la raíz sea de los mas chicos el mas grande, o el mas chico de los mas grandes

Barridos de arboles: como se leen los arboles

* Preorden: el nodo se lee apenas se llega al mismo
* Postorden: el nodo se lee cuando se va del mismo y no se va regresar
* Inorden: el nodo se lee cuando se cambia de rama en el árbol

Preorden Inorden

*if root if root*

*printf(root->dato); inorden(root->izq); preorden(root->izq); printf(root->dato); preorden(root->der); inorden(root->der);*

*return; return;*

*\*

Arbol de expression: puedo resolver una expresión si lo meto en un arbol

Si a dicho árbol se barre inorder se obtiene la expresión matemática en notación INFIJO, si se lo barre postorden se obtiene la expresión matemática en notación POSTFIJO o POLACA INVERSA

A diagram of a network

AI-generated content may be incorrect.

INORDEN: 3 + 5 \* 8 – 4 \* 2

PORTORDEN: 358\*+42\*-