**ARBOLES**

**Contenido**

* **Fundamentos teóricos**
* **Tipos de arboles**
* **Arboles binarios de búsqueda**
* **Implementación de operaciones básicas de árboles binarios**
* **Implementación de operaciones complementarias de árboles binarios**
* **Aplicaciones de árboles binarios de búsqueda**

**Fundamentos teóricos**

Las estructuras dinámicas lineales de datos listas enlazadas, pilas y colas tienen grandes ventajas de flexibilidad sobre las representaciones contiguas, sin embargo, tienen un punto débil, son listas secuenciales, es decir, están dispuestas de modo que es necesario moverse a través de ellas una posición cada vez (cada elemento tiene un siguiente elemento).

Las estructuras de datos que resuelven los problemas que plantean las listas lineales y en las que cada elemento puede tener diferentes “siguientes” elementos, que introducen el concepto de estructura de bifurcación. Estos tipos de datos se llaman árboles.

El árbol es una estructura de datos muy importante en informática y en ciencias de la computación. Los árboles son estructuras no lineales, al contrario que los arrays y las listas enlazadas que constituyen estructuras lineales.

Los árboles son muy utilizados en informática para representar formulas algebraicas, como un método eficiente para búsquedas grandes y complejas en listas dinámicas y en aplicaciones diversas tales como inteligencia artificial.

Intuitivamente el concepto de árbol implica en la que los datos se organizan de modo que los elementos de información están relacionados entre sí a través de ramas. El árbol genealógico es el ejemplo típico más representativo del concepto de árbol general.

* Un árbol consta de un conjunto finito de elementos, denominados nodos, y un conjunto finito de líneas dirigidas, denominadas ramas, que conectan los nodos. El número de ramas asociado como un nodo es el grado del nodo.
* Un árbol es un conjunto de uno o más nodos tales que: hay un nodo diseñado especialmente llamado raíz.
* Si un árbol no está vacío, entonces el primer nodo se llama raíz. Obsérvese en la definición 2 que el árbol ha sido definido de modo recursivo ya que los subárboles de definen como árboles.

Un árbol se puede definir como una estructura jerárquica y en forma no lineal, aplicada sobre una colección de elementos u objetos llamados nodos. (Cairó & Guardali, 2006).

**Árboles**

Se les llama estructuras dinámicas, porque las mismas pueden cambiar tanto de forma como de tamaño durante la ejecución del programa.

Y estructuras no lineales porque cada elemento del árbol puede tener más de un sucesor.

|  |  |
| --- | --- |
| Estructuras estáticas | Estructuras dinámicas |
| Arreglos | Listas |
| Registros | Arboles |
|  | Graficas |

|  |  |
| --- | --- |
| Estructuras lineales | Estructuras no lineales |
| Arreglos | Arboles |
| Registros | Graficas |
| Pilas |  |
| Colas |  |
| Listas |  |

**Características y terminología de los arboles**

La representación y terminología de los árboles se realiza con las notaciones de las relaciones familiares genealógicas: padre, hijo, hermano, ascendente, descendiente, etc.

Las estructuras de tipo árbol se usan principalmente para representar datos con una relación jerárquica entre sus elementos, tales como son árboles genealógicos, tablas, etc.

Un árbol A es un conjunto finito de uno o más nodos, tales que:

1. Existe un nodo especial denominado RAIZ(v1) del árbol.
2. Los nodos restantes A1, A2, …, Am, cada uno de los cuales es, a su vez, un árbol. Estos árboles se llaman subárboles del RAIZ.

En relación con otros nodos

**Nodos:** se le llama nodo a cada elemento que contiene el árbol.

**Nodo padre:** se utiliza este término para llamar aquellos nodos que tienen al menos un hijo.

**Nodo hijo:** los hijos son todos aquellos nodos que tienen un padre.

**Nodo hermano:** los nodos hermanos son aquellos nodos que comparten un mismo padre en común dentro de la estructura.

**En relación con la posición dentro del árbol**

**Nodo raíz:** se refiere al primer nodo de un árbol, solo un nodo del árbol puede ser la raíz.

**Nodo hoja:** son todos aquellos nodos que no tienen hijos, los cuales siempre se encuentran en los extremos de la estructura.

**Nodo interior o rama:** estos son todos aquellos nodos que no son la raíz y que además tiene al menos un hijo.

**En relación con el tamaño del árbol**

**Nivel:** el nivel del nodo es su distancia a la raíz. Por lo tanto:

* Un árbol vacío tiene 0 niveles
* El nivel de la raíz es 1
* El nivel de cada nodo se calcula contando cuantos nodos existen sobre el, hasta llegar a la raíz + 1, y de forma inversa también se podría contar cuantos nodos existen desde la raíz hasta el nodo buscando + 1.

**Altura:** se le llama altura al número máximo de niveles de un árbol.

**Con relación al tamaño del árbol**

**Peso:** es el número de nodos que tiene un árbol.

**Orden:** el orden de un árbol es el número máximo de hijos que puede tener un Nodo. Es una constante que se define antes de crear el árbol.

Este valor no se calcula, sino que ya se conoce cuando se diseña la estructura.

**Grado:** número de hijos de un nodo y está limitado por el orden, ya que este indica el número máximo de hijos que puede tener un nodo.

**El** **grado de un árbol** se define como el máximo grado de todos sus nodos.

**Camino:** secuencia de nodos conectados dentro del árbol.

**Longitud del camino:** cantidad de nodos que se deben recorrer para llegar desde la raíz a un nodo determinado.

**Subárbol:** conocemos como sub-arbol a todo árbol generado a partir de una sección determinada del árbol. Por loque podemos decir que un árbol es un nodo raíz con N sub-arboles.

**TIPOS DE ÁRBOLES**

Los árboles pueden clasificarse tomando en cuenta su estructura y funcionamiento. A continuación, se presentan los tipos de árboles más utilizados:

Arboles binarios:

* Arboles binarios distintos
* Arboles binarios similares
* Arboles binarios equivalentes
* Arboles binarios completos
* Aboles binarios llenos
* Arboles binarios degenerados
* Arboles binarios de búsqueda
* Arboles equilibrados

Arboles multicaminos:

* Arboles -B
* Arboles B+
* Arboles 2-4

**Arboles binarios**

Esta estructura se caracteriza porque cada nodo puede tener máximo 2 hijos, dicho de otra manera, es un árbol de dos grados.

Un árbol binario es un árbol en el que ningún nodo puede tener mas de dos subárboles. En un árbol binario, cada nodo puede tener cero, uno o dos hijos (subárboles). Se conoce al nodo de la izquierda como hijo izquierdo y el nodo de la derecha como hijo derecho.

Un árbol binario no puede tener más de dos subárboles.

Un árbol binario es una estructura recursiva. Cada nodo es la raíz de su propio subárbol y tiene hijos, que son raíces de árboles llamados los subárboles derecho e izquierdo del nodo, respectivamente.

Un árbol binario se divide en 3 subconjuntos disjuntos:

* {R} Nodo raíz
* {11,12,…In} subárbol izquierdo de R
* {D1,D2,…Dn} subárbol derecho de R

**Arboles binarios distintos, similares y equivalentes**

**Arboles binarios distintos:** dos árboles binarios son distintos cuando sus estructuras son diferentes.

**Arboles binarios similares:** dos árboles binarios son similares cuando sus estructuras son idénticas, pero la información que contienen sus nodos difiere entre sí.

**Arboles binarios equivalentes:** los árboles binarios equivalentes se definen como aquellos que son similares y además los nodos contienen la misma información.

**Arboles binarios completos**

Un árbol binario completo de profundidad n es un árbol en el que, para cada nivel, del 0 al nivel n-1 tiene un conjunto lleno de nodos y todos los nodos hoja a nivel n ocupan las posiciones más a la izquierda del árbol.

**Arboles binarios llenos**

Es un árbol lleno donde todos los nodos tienen cero o dos hijos. Es decir, no existe un nodo que tenga un solo hijo.

**Arboles binarios degenerados**

Es un tipo especial denominado árbol degenerado en el que hay un solo nodo hoja y cada nodo no hoja solo tiene un hijo.

Un árbol degenerado es equivalente a una lista enlazada.

**Arboles binarios equilibrados**

Cuando un árbol binario de búsqueda crece descontroladamente hacia un extremo su rendimiento puede disminuir consideradamente.

Para mantener la eficiencia de operación surgen los árboles equilibrados o balanceados.

Estos pueden realizar acomodos o balanceos después de inserciones o eliminaciones de elementos.

**Arboles binarios perfectamente equilibrados**

Un árbol perfectamente equilibrado es un árbol binario en el que, para todo nodo, el número de nodos en el subárbol izquierdo y el número de nodos en el subárbol derecho difieren como mucho en una unidad.

**Arboles binarios equilibrados AVL**

un árbol equilibrado en sentido AVL (Adelson-Velskii y Landis, 1962) es un árbol binario en el que la diferencia de alturas de los subárboles izquierdo y derechos correspondientes a cualquier nodo del árbol no es superior a uno.

**Arboles multicaminos**

Un árbol multicaminos es una estructura de datos homogénea, dinámica y no lineal, en donde a cada nodo le pueden seguir una cantidad n de nodos hijos.

Mientras que los árboles binarios fueron pensados para trabajar en memoria principal, los árboles multicaminos fueron diseñados para trabajar con sistemas de archivos (Serrano Montero, 2006).

**Arboles binarios de búsqueda**

Un árbol binario que tiene los nodos ordenados de alguna manera se conoce como árbol binario de búsqueda.

Se puede buscar aplicando el criterio de búsqueda similar al utilizado con arreglos.

En concreto, un árbol binario de búsqueda es aquel en que, todos los datos del subárbol izquierdo son menores que los datos de ese nodo, mientras que todos los datos del subárbol derecho son mayores que el nodo.

Formalmente se define un árbol binario de búsqueda de la siguiente manera (Cairó & Guardati, 2006):

Para todo nodo T del árbol se debe cumplir que todos los valores almacenados en el subárbol izquierdo de T sean menores a la información guardada en el nodo T. de forma similar, todos los valores almacenados en el subárbol derecho de T deben ser mayores a la información guardada en el nodo T.

Es una estructura de datos sobre la cual se pueden realizar eficientemente las operaciones de búsqueda, inserción y eliminación.

Comprobando con otras estructuras de datos, se pueden observar ciertas ventajas:

En un arreglo es posible localizar datos eficientemente si estos se encuentran ordenados, pero las operaciones de inserción y eliminación resultan costosas, porque involucran movimiento de los elementos dentro del arreglo.

Es las listas, por otra parte, dichas operaciones se pueden llevar a cabo con facilidad, pero la operación de búsqueda, en este caso es una operación que demanda recursos, pudiendo inclusive requerir recorrer todos los elementos de ella para llegar a uno en particular.

**Creación de un árbol binario de búsqueda**

Siguiendo la regla, dado que un nodo en el árbol todos los datos a su izquierda deben ser menores que todos los datos del nodo actual, mientras que todos los datos a la derecha deben ser mayores que los datos.

En los árboles binarios no puede haber datos repetidos.

**Implementación de árboles binarios de búsqueda**

Los árboles binarios pueden ser representados de dos modos diferentes:

* Mediante punteros (lenguajes C y C++)
* Mediante arrays o listas enlazadas

Vinculando nodos, objetos con miembros que referencian otros objetos del mismo tipo.

**Operaciones para implementar:**

1. Constructor
2. Destructor
3. esVacio()
4. Insertar elemento
5. Mostrar árbol completo acostado con la raíz a la izquierda\*
6. Graficar árbol completo
7. Buscar un elemento en el árbol
8. Recorrer el árbol en PreOrden
9. Recorrer el árbol en InOrden
10. Recorrer el árbol en PostOrden
11. Eliminar un nodo del árbol PREDECESOR
12. Eliminar un nodo del árbol SUCESOR

**Operaciones complementarias:**

1. Recorrer el árbol por niveles (Amplitud)
2. Altura del árbol
3. Cantidad de hojas del árbol
4. Cantidad de nodos del árbol
5. Revisa si es un árbol binario completo
6. Revisa si es un árbol binario lleno
7. Eliminar el árbol

Para analizar las operaciones de un apila se deben crear tres elementos que permitan implementar el árbol:

**Clase nodo:** esta clase representa un nodo individual en la estructura de datos y contiene información sobre el valor almacenado en el nodo y punteros al nodo izquierdo y derecho.

**Clase árbol:** esta clase representara las operaciones que se pueden realizar con las pilas.

**Método main:** método principal en donde se creará una instancia del árbol para acceder a sus propias operaciones.

**Recorrido en InOrden**

Este recorrido se realiza así: el primero recorre el subárbol izquierdo, el segundo visita la raíz y por último, va al subárbol derecho, en síntesis (izquierdo, raíz, derecho). El valor en un nodo no se procesa hasta que se procesen los valores en su subárbol izquierdo.

Para recorrer un árbol binario no vacío en **InOrden**, hay que realizar las siguientes operaciones recursivamente en cada nodo, comenzando por el nodo raíz:

* Recorrer el subárbol izquierdo **InOrden**
* Procesa la raíz
* Recorrer el subárbol derecho **InOrden**

**Recorrido en PreOrden**

Este recorrido se realiza así: primero visita la raíz; segundo recorre el subárbol izquierdo y por último va al subárbol derecho, en síntesis (raíz, izquierdo, derecho). Para recorrer un árbol binario no vacío en **PreOrden**, hay que realizar las siguientes operaciones recursivamente en cada nodo, comenzando con el nodo raíz:

* Procesa la raíz
* Recorrer el subárbol izquierdo en **PreOrden**
* Recorrer el subárbol derecho en **PreOrden**

**Recorrido en PostOrden**

Primero recorre el subárbol izquierdo; segundo, recorre el subárbol derecho y por último visita la raíz, en síntesis (izquierdo, derecho, raíz). Para recorrer un árbol binario no vacío en **PostOrden**, hay que realizar las siguientes operaciones recursivamente en cada nodo, comenzando con el nodo raíz:

* Recorrer el subárbol izquierdo en **PostOrden recorrer**
* El subárbol derecho en **PostOrden** procesa la raíz.

Tarea: implementar los 3 recorridos de arboles en equipo para el próximo martes 21 de noviembre.