**Implementación de árboles de manera poco convencional**

Los árboles son generalmente implementados a través de enlaces de nodos, compuestos de dos punteros que almacenarán a otros nodos o valores nulos, una filosofía muy parecida a la implementación de una estructura lineal dinámica. Sin embrago, para representar ciertos problemas, podemos abstraer los árboles de otra manera tal q, que nos permita representarlo de distinta forma a la convencional. Aquí algunos ejemplos:

**-Implementación mediante una lista de listas:** Es útil para problemas que requieran un árbol de soluciones, como en el caso de llevar el rastreo de la ruta que se toma en un método de búsqueda.

En esta implementación, cado nodo apunta a una lista enlazada, los nodos de dicha lista podrán tener a otro nodo enlazado que contiene a su vez, otra lista enlazada (cada nodo apunta a un nodo hijo).

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**-Implementación como un array de elementos del tipo base**: A través de un array, podemos almacenar los datos siguiendo un orden establecido. El primer elemento del array representará al nodo raíz, y los siguientes nodos se irán guardando en los lugares nodoPadre+1 y nodoPadre+2, a partir de ahí, nuestro array crecerá el doble cada que agreguemos un nuevo nivel de profundidad en el árbol.

Interfaz de usuario gráfica, Texto, Aplicación, Word

Descripción generada automáticamente

**-Implementación como un array de nodos:** Empleando una matriz con 3 componentes para las columnas y n componentes para las filas, podemos almacenar valores que nos representen los enlaces existentes de un árbol. La columna 0 representará el índice del hijo izquierdo del nodo, la columna 2 será el índice del hijo derecho, y la columna 1(columna de en medio) nos representa la información del nodo en cuestión.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**Árboles de decisión**

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Encuentran las posibles soluciones basadas en condiciones prestablecidas, y es muy empleado en machine learning para tareas de clasificación o regresión.

Los árboles de decisión tienen un primer nodo llamado raíz (root) y luego se descomponen el resto de los atributos de entrada en dos ramas (podrían ser más, pero no nos meteremos en eso ahora) planteando una condición que puede ser cierta o falsa. **Se bifurca cada nodo en 2 y vuelven a subdividirse hasta llegar a las hojas que son los nodos finales** y que equivalen a respuestas a la solución: Si/No, Comprar/Vender, o lo que sea que estemos clasificando

Aprendzaje inductivo: Árboles de decisión

se basa en el descubrimiento de patrones a partir de ejemplos.

Suponiendo que tenemos N ejemplos que vienen definidos a partir de un conjunto de atributos(propiedades), tenemos una clasificación prestabelcida

La tarea del aprendizaje inductivo consiste en inducir de los datos anteriores un mecanismo para inferir las clasificaciones de cada uno de los ejemplos a partir únicamente de las propiedades.

odríamos predecir comportamientos futuros a partir de los comportamientos observados en el pasado. Por ejemplo, Otro ejemplo que es muy usado en la actualidad se encuentra en el mundo de los créditos bancarios, a partir de los comportamientos de los clientes antiguos con respecto a la morosidad o no de sus pagos del crédito concedido, podemos inferir qué nuevos clientes tienen más probabilidad de hacer frente al pago del mismo y cuáles más probabilidad de dejarlo sin pagar.

árboles de decisión, que proporcionan un conjunto de reglas que se van aplicando sobre los ejemplos nuevos para decidir qué clasificación es la más adecuada a sus atributos/propiedades.

Un árbol de decisión está formado por un conjunto de nodos de decisión (interiores) y de nodos-respuesta (hojas):

Un nodo de decisión está asociado a uno de los atributos y tiene 2 o más ramas que salen de él, cada una de ellas representando los posibles valores que puede tomar el atributo asociado. De alguna forma, un nodo de decisión es como una pregunta que se le hace al ejemplo analizado, y dependiendo de la respuesta que de, el flujo tomará una de las ramas salientes.

Un nodo-respuesta está asociado a la clasificación que se quiere proporcionar, y devuelve la decisión del árbol con respecto al ejemplo de entrada.

**Variación del Árbol B:**

**Árboles B+**

Nos proporcionan la misma ventaja de contar con un acceso aleatorio y conservan la propiedad del recorrido secuencial rápido, pero todas las claves se encuentran en las hojas, duplicándose en la raíz.

Estos árboles ocuparán más espacio debido a la propiedad de tener duplicadas algunas claves de información, las claves de las páginas raíz e interiores se utilizarán únicamente como índices.

Todas las claves se encuentran en hojas, duplicándose en la raíz y nodos interiores aquellas que resulten necesarias para definir los caminos de búsqueda. Para facilitar el recorrido secuencial rápido las hojas se pueden vincular, obteniéndose, de esta forma, una trayectoria secuencial para recorrer las claves del árbol.

En el método de inserción, el árbol B y B+ cambian, debido a que cuando se inserta una nueva clave en una página llena, ésta se divide en otras dos y lo que subirá a la página padre será una copia. Las llaves se almacenan en los nodos del último nivel del árbol y esos nodos se enlazan unos con otros formando una lista ligada que puede estar enlazada de forma sencilla o doblemente.

Al dividirse las páginas, la primera hoja contendrá (tomando a m como el orden) m/2 claves y la segunda (m/2) +1, y lo que subirá a la página antecesora será una copia de la clave central.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Para el borrado, habrá que considerar:

* Si al eliminar la clave (siempre en una hoja) el número de claves es mayor o igual a m/2 el proceso ha terminado. Las claves de las páginas raíz o internas no se modifican, aunque sean una copia de la eliminada, pues siguen constituyendo un separador válido entre las claves de las páginas descendientes.
* Si al eliminar la clave el número de ellas en la página es menor que m/2 será necesaria una fusión y redistribución de estas tanto en las páginas hojas como en el índice.

**Ventaja de los árboles Bayer vs los AVL en términos de memoria**

Es cierto que para un problema que requiera el manejo mayor volumen de datos, los árboles B proporcionan una herramienta más eficaz para el acceso de estos. Pero ¿Cómo interpretamos esto en términos de acceso a memoria?

En memoria interna el tiempo de acceso a n datos situados en distintas partes de la memoria es independiente de las direcciones que estos ocupen, mientras que en memoria externa es fundamental el acceder a datos situados en el mismo bloque para hacer que el tiempo de ejecución disminuya, si disminuimos el número de accesos a disco, lógicamente el tiempo resultante de ejecución de nuestra búsqueda se ve fuertemente recortado

Entonces, debido a que el tiempo de ejecución se incrementa junto con el número de acceso requeridos, un árbol binario requerirá un mayor número de accesos a disco para cargar un nodo en memoria y decidir entre dos posibles ramas. Mientras que un árbol con múltiples ramas y múltiples elementos en cada nivel nos proporciona la posibilidad de leer un número alto de datos sin necesidad de acceder a otro nodo

Posibles temas:

Árboles de expansión

Árboles de expresión

**Otros algoritmos de búsqueda en grafos**

* **Algoritmo Breadth-First Search**
* **Algoritmo de Bloques**
* **Algoritmo de Dijkstra**