Ing. de Sistemas e Informática
Estructuras de Datos y Algoritmos
Jorge Mario Londoño P.

Taller 4

Árboles binarios de búsqueda

Introducción

Típicamente las aplicaciones necesitan "indexar" conjuntos de datos sobre distintos atributos con el fin de realizar consultas. Para tal fin se pueden crear árboles binarios de búsqueda sobre los atributos que interese considerar, pero resultaría muy ineficiente duplicar todos los datos tantas veces como registros se tienen, por lo que un índice simplemente es una estructura separada que contiene solo referencias a los datos indexados y múltiples índices pueden hacer referencia al mismo conjunto de datos.

En este taller se exploran las técnicas para indexar conjuntos de datos y realizar operaciones utilizando estos índices.

Ejercicio a desarrollar

1. El archivo `appl_stock.csv` es un archivo separado por comas que contiene datos ejemplo para este ejercicio. Definir el ADT `ValoresFecha` que representa los valores de una acción para una fecha determinada. Definir un constructor que tome como entrada una línea del archivo (`String` separado por comas) y cree una instancia del `ValoresFecha`. Usar tipos referencia `Comparable` para todos los datos que componen un registro. Desarrollar un método auxiliar estático que tome como argumento la ruta/url del archivo y retorne un Bag<ValoresFecha> con la colección de datos contenidos en el archivo. Sobre-escribir el método `toString` para poder obtener la descripción de un ValorFecha organizado para impresión.

Implementar una clase `Taller4` que contendrá el `main` del programa, las estructuras auxiliares a crear y las consultas que se solicitan en los puntos siguientes:

2. Crear tres índices sobre la colección de datos: Uno por fecha, otro por valor de apertura (open) y otro por volumen. Los indices serán variables estáticas de tipo BST<Key,ValoresFecha>, donde Key es el tipo de la respectiva columna. Importante: Los valores son referencias a los objetos `ValoresFecha` que estan en el Bag, para no duplicar la información. Implementar tres métodos `indexarPorXXX` para crear los índices.

3. Crear un método `consultaPorFecha` que tome como argumentos dos fechas e imprima en pantalla los ValoresFecha comprendidos en ese rango de fechas.

4. Crear un método `consultaCuartilesValor` que obtenga los valores de apertura (Open) correspondientes al mínimo, primer cuartil (25%), segundo cuartil (50%), tercer cuartil (75%) y máximo. El valor de cada cuartil es el valor tal que el x% de los datos son menores a ese valor. Los resultados de cada cuartil se imprimen en pantalla.

5. Obtener los volúmenes correspondientes a los rankings 1..10 de mayor a menor (en otras palabras los registros correspondientes a los 10 mayores volúmenes) en un método `rankingsPorVolumen`. Para cada ranking imprimir una línea con la siguiente información: Ranking, fecha, apertura, volumen, valor-total=apertura*volumen.

6. Crear un método main que invoque y todos los procesos anteriores:

- Cargar los datos desde una ruta/url especificado por medio de los argumentos.

- Crear los indices sobre los tres atributos indicados.

- Realizar un ejemplo de la `consultaPorFecha`.

- Realizar un ejemplo de la `consultaCuartilesValor`.

- Realizar un ejemplo de la consulta `rankingsPorVolumen`.

Opcionales

Para cada una de las consultas de los numerales 3, 4 y 5 estimar su orden de crecimiento utilizando como modelo de costo el número de comparaciones necesarias. (+0.5 por c/u).

Entregables

Las dos clases que componen el programa (`ValoresFecha.java, Taller4.java`). Documento escrito para los puntos opcionales.

En caso de utilizar estructuras de las bibliotecas del texto (`algs4.jar`) **no** anexar la biblioteca.

Grupos máximo de 2 personas.