Universidad de La Serena
Plan General

Diseño y Análisis de Algoritmos
Profesor: Erick Luciano Castillo Bastias
Fecha : Miércoles 29 de mayo
Equipo 2

Integrantes :
Fernando López
Rodrigo Mamani
Bryan Townsend

Índice

1. Introducción

Se debe realizar un software que ordene una secuencia de datos utilizando los algoritmos: InsertionSort, MergeSort y QuickSort. Además implementar los métodos array.sort() y collection.sort() (Los métodos utilizados por Java para ordenar arreglos y listas respectivamente). El software deberá tener una GUI que permita generar números aleatorios, números ordenados y permita mostrar el estado actual de los datos almacenados en memoria principal. Se debe hacer distintas pruebas cuando el software esté implementado estas son: Datos del orden entre 1 y 6 órdenes de magnitud. Los algoritmos de ordenamiento para este proyecto son basado en comparación, por tanto tienen un costo en el peor caso como máximo $\Omega(nlogn)$. Esto se observa para MergeSort. En el caso de QuickSort este algoritmo tiene costo en el peor caso de $O(n^2)$, esto ocurre cuando los elementos ya están ordenado. InsertionSort tiene costo en el caso promedio de $O(n^2)$ y en el mejor caso de $O(n)$ cuando los datos ya están almacenados. Este proyecto nos permitirá verificar estos resultados no solo mediante la implementación de la GUI, sino que la implementación de los códigos en un lenguaje como Java, nos mostrará como se van sumandos los distintos costos y por qué tienen estos costos.

Como corresponde a un laboratorio del curso diseño y análisis de algoritmo de la carrera de Ingienería en computación de la universidad de La Serena nos enfocaremos en explicar y señalar cómo implementar estos algoritmos, con ello comparando a los métodos utilizados por java para ordenar.

Se divide la implementación en insertionSort y métodos array.sort y collection.sort para un desarrollador y la implementación de mergeSort y QuickSort para otro desarrollador. El trabajo se debe efectuar en 6 días. En 4 días debemos tener las implementaciones de los 3 algortimos y la documentación de este plan general y de un informe con los requerimientos, utilizando los modelos vistos en la clase de laboratorio. En los dos últimos días se deben realizar la GUI y testear nuestro software.

2. Ámbito

Software desarrollado para un usuario. Visualizar datos ordenados en una GUI. Utilizando datos aleatorios y datos ordenados para poder tener una aproximación heurística de los costos de los algoritmos. Este software es llevado a cabo como parte de una tarea de un curso de laboratorio de diseño y análisis de algoritmos por tanto su ámbito esta basado en un usuario exclusivo (el profesor). El tiempo máximo de respuesta no debe superar el orden de horas para el caso de datos de ordenes de magnitud 6, esto porque consideramos que en el peor caso tendremos un tiempo asintótico de $O(n^2)$. La interfaz estará basada en frames de la librería Swing de Java y permitirá al usuario elegir en: tipo de datos generados (números aleatorios u ordenados) y algoritmo de ordenamiento(insertionSort, mergeSort o quickSort). La fiabilidad del software es simple, ya que al ser una tarea no requerimos de datos personales del usuario.

3. Alcance

El alcance de este programa es poder implementar las tres operaciones asociadas a las estructuras de datos por medio de una aplicación hecha en Java. Esto va dirigido a usuarios con conocimiento de estructuras de datos jeraraquizados, ya que presentamos distintas caracteristicas de nuestro árbol (altura, número de nodos, elementos). Para ello utilizaremos la libreria Swing de Java para poder visualizar esta instancia del árbol y la versión Java 1.8

4. Restricciones

Nuestro programa presenta las caracteristicas de las estructuras de datos, pero no la representación visual del árbol. No presenta elementos en otro tipo de dato que sea entero. Nuestro árbol B-Tree se realiza en memoria principal, los elementos del árbol no se modifican cuando el usuario seleccione distintos tipos de árbol. El programa no permitirá elementos repetidos, esto significará no usar genericos.

5. Meta

La meta de nuestro software es visualizar en una GUI: Altura actual, número de elementos, número de nodos del árbol a medida que el usuario vaya incorporando elementos para los árboles ABB, AVL y B-Tree

6. Objetivos

La planificación consistirá en estos 4 dias en dedicar al análisis, los 3 primeros dias al analizar el problema y los últimos 2 dias a la implementación del código. Y para realizar las pruebas el día anterior (los 20000 datos). Será un trabajo colaborativo, pero con mayor énfasis en las estructuras de datos ABB y AVL para un desarrollador y otro desarrollador para el árbol B-Tree. Esperamos utilizar genéricos y para el caso de B-Tree poder dejar este árbol en memoria secundaria, para ello deberíamos implementar un archivo que almacene la raiz. El objetivo principal(el cual es el de la asignatura) es implementar esta aplicacion como un proyecto de software, entender la documentación y los procesos que requiere. Revisar los distintos estándar de la IEEE (830 Requerimientos, 730 Plan de aseguramiento de calidad, 1063 marco de referencia que debe tener cualquier documento destinado a usuarios). El desarrollo será de tipo cascada, ya que como es un programa pequeño a medida que vamos desarrollando la aplicación, vamos revisando los errores, no nos devolvemos para corregir (Esto por el tiempo elegido para organizar este trabajo).