Juan Alegría, Andrés Molina, Alejandra Vargas

202011282 - 201923434 - 201123148

Docente: Dario Correal

Estructura de Datos y Algoritmos - Sección 2 - Equipo 4

23 de agosto de 2020

Laboratorio

Fuente de Datos	Arraylist [ms]	Singlelinkedlist [ms]
Peliculas (MoviesDetailsCleaned)	31.250	78.125
Elenco (MoviesCastingRaw)	62.500	62.500

Tabla 1. Tabla de rendimiento para cargar archivos como lista enlazada y arreglo.

Algoritmo	Arraylist [ms]	Singlelinkedlist [ms]
SelectionSort	2906.250	266656.250
InsertionSort	2094.755	302441.601
ShellSort	15.625	1484.375

Tabla 2. Tabla de rendimiento para los distintos algoritmos de ordenamiento en listas enlazadas y arreglos.

Link al repositorio: https://github.com/EDA-SEC-02-EQUIPO-04/Lab2_202020

Preguntas

- ¿Qué diferencias se observan en el desempeño de la carga de datos entre arreglo (*Arraylist*) y lista sencillamente encadenada (*SingleLinkedlist*)?

  R// Por lo general, el arreglo suele ser mucho más rápido que las listas encadenadas, respecto a estas operaciones iniciales y carga de datos. Posiblemente, esto es debido al espacio y tiempo que tienen que gastar las listas encadenadas al configurar y encadenar punteros a los siguientes elementos, mientras que el arreglo no tiene la necesidad de realizar esto y esta podría ser la principal causa por la cual es más rápida la lectura y escritura de datos.

- ¿Cuál de las dos implementaciones (*Arraylist* y *SingleLinkedlist*) tiene mejor desempeño? Y ¿Por qué?

  R// Entre las dos implementaciones ninguna tiene mejor o peor desempeño, ya que esto está estrechamente unido a las operaciones y necesidades del programa o usuario. Por esta misma razón, es que implementamos estructuras de datos abstractas y así tenemos la posibilidad de modificar, probar y utilizar fácilmente varios tipos de implementaciones sin tener que realizar grandes cambios en el código.

- ¿Qué diferencias existen entre cargar los archivos de películas (*MoviesDetailsCleaned*) y elenco (*MoviesCastingRaw*)?, ¿Por qué se presentan estas diferencias?

  R// Para MoviesDetailsCleaned se tiene un tiempo de 8.890625 segundos, mientras que para MoviesCastingRaw se tienen 6.59375 segundos. Esto se puede deber a que

aunque se tiene la misma cantidad de columnas (329044) de columnas para el de las películas se tienen 22 y del casting se tienen 19, suministrando así una mayor cantidad de datos en el primero.

- ¿Qué diferencias en el desempeño se observan entre los tres algoritmos de ordenamiento?

  R//

  - **Selection sort:** El movimiento que tiene que hacer con los datos es mínimo ya que trabaja con dos entradas en las cuales utiliza N intercambios, sin embargo, mientras encuentra el menor o mayor elemento no da información sobre estos hasta que sigue con el siguiente. En este caso con los datos de la tabla 2 se evidencia que fue el segundo en tener un menor tiempo.
  - **Insertion sort:** El tiempo que se demora depende del orden inicial en que estén los datos, a su vez, este puede demorarse más debido a que solo cambia un dato a la vez. En este caso ya que los datos no estaban parcialmente ordenados por lo que le tomó más tiempo llegar a la solución.
  - **Shellsort:** Como se observa en la tabla 2, este algoritmo fue el de mejor desempeño, esto se puede deber a que puede comparar más de un elemento a la vez, ordenando así de forma más eficiente los datos.

- ¿Qué efectos tienen los dos tipos de lista en los tres algoritmos de ordenamiento?

  R// En los tres algoritmos de ordenamiento sigue siendo mucho más rápido hacer uso de arreglos que de listas sencillamente encadenadas. Siendo de este modo que los algoritmos de ordenamiento se demoran cerca de cien veces más en las listas encadenadas que en los arreglos.