Trabajo Práctico Especial
Programación 3
TUDAI
Primera Entrega
(con correcciones)

Facultad de Ciencias Exactas
Universidad Nacional del Centro

Alumno: Sandoval, Fernando Daniel
Email: fdsandoval@gmail.com
Fecha de entrega: 4/5/2018
Fecha de entrega versión corregida: 6/6/2018

Introducción

La cátedra nos solicita que implementamos una herramienta que permita simplificar la busqueda de libros por géneros. Y además caracterizar el comportamiento de los usuarios mientras realizan dichas busquedas. En una primera etapa se desea implementar la logica para obtener una colección de libros que contenga un género en particular. Para simplificar el proceso de búsqueda se requiere implementar un índice por género. Básicamente, un indice es una estructura que mejora sustancialmente la velocidad con la que se acceden a los datos de una tabla, y su efectividad se ve reflejada cuando se manejan grandes volúmenes de datos. En el caso de una búsqueda por ejemplo, sólo hay que buscar en el índice dicho elemento para, una vez encontrado, devolver un registro que se encuentre en la posición marcada por el índice, en lugar de tener que buscarlo secuencialmente, con el costo que esto significa.

Análisis

La catedra nos brinda inicialmente una serie de archivos de datos (datasets) de diverso tamaño, para comenzar a realizar nuestro trabajo. Se analizaron previamente diversas posibles estructuras para la implementación, e inicialmente se optó por utilizar una lista de listas para volcar los datos de los archivos .csv en memoria y a partir de allí generar los indices de acceso por género. Luego se detectó que podía generarse el indice directamente desde la lectura de los archivos .csv, sin necesidad de una estructura auxiliar, con lo cual nos ahorramos el tiempo de carga de la

estructura de datos en memoria, y tambien la memoria utilizada para dicha estructura. Un segundo análisis nos llevó a tener que decidir entre tres estructuras posibles:

- ArrayList
- Clase Lista (previamente implementada en la materia)
- Arbol Binario

Para tomar una decisión se llevaron a cabo comparativas entre las tres estructuras en cuanto a tiempos de acceso y complejidad. En el caso de un ArrayList, la inserción de un elemento tiene un costo de O(N) con tamaño del array igual a N, mientras que la busqueda se hace directamente sobre el indice, con lo cual su costo es de O(1). La inserción en la lista vinculada tiene un costo de O(1) ya que se realiza al final de la misma (o al principio, sgeun su implementación), mientras que el borrado tiene costo O(N) en el peor caso ya que hay que recorrer uno por uno los elementos hasta encontrar el que queremos borrar. Por ultimo, un arbol binario de busqueda tiene un costo de inserción de O (N log n) con N elementos a insertar, y la búsqueda en el arbol tiene en promedio un costo similar, siendo a priori ésta la opción más eficiente.

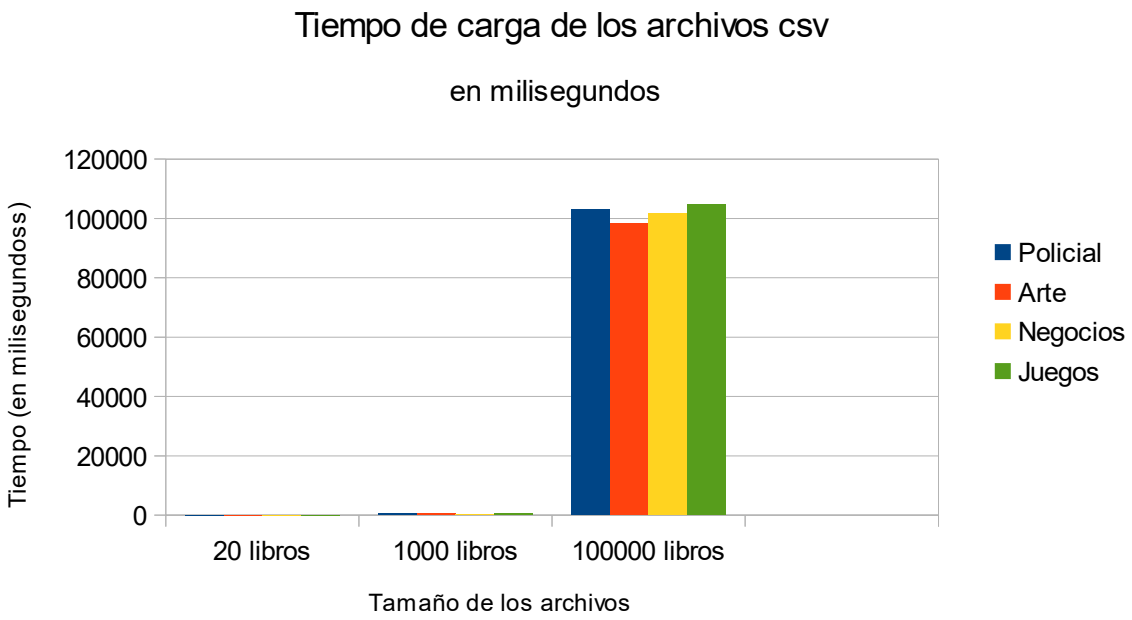
Toma de decisiones e implementación

Para decidir que estructuras utilizar, se tuvo en cuenta elegir el caso promedio entre eficiencia y facilidad de implementación, para finalmente decidirnos por una lista vinculada. La velocidad de ejecución de la herramienta dependerá linealmente del número de elementos de la lista. Para la implementación de la herramienta se utilizó una lista cuyos nodos tienen cada uno el nombre del género y una lista de libros pertenecientes a ese género. La lista de generos se va creando a medida que se va recorriendo el archivo csv, como asi tambien la lista de los libros que

pertenecen a cada género. Una vez creada, facilmente pueden obtenerse los datos correspondientes a los títulos de los libros que pertenecen a un determinado género, haciendo una busqueda simple en la lista, pasándole como parámetro un String con el género buscado.

Mediciones

Se tuvo en cuenta para realizar las distintas mediciones el ingreso por teclado del género a ser buscado en el indice, ya que es lo único que puede hacer variar en mayor o menor medida el tiempo de ejecución. Los tiempos de lectura de los datos desde el archivo csv sólo varían dependiendo del tamaño del archivo csv. Tomamos diferentes muestras de tiempos y las comparamos a continuación:



	Policial	Arte	Negocios	Juegos
20 libros	31	38	91	28
1000 libros	560	480	305	528
100000 libros	103145	98350	101547	104637

En la muestra de datos vemos que la carga de 20 libros en promedio duplica la cantidad de accesos, mientras que para 1000 se tiene el caso promedio (n/2), y finalmente para 100000 se obtiene un número similar al del tamaño de la lista. No se han considerado en esta tabla los valores correspondientes a la carga de 1000000 de libros dado que se obtuvieron valores muy altos, del orden de los 10000000 de milisegundos.

Conclusiones

Como podemos ver, la estructura utilizada nos permitió una rápida implementación con el objetivo de realizar las pruebas necesarias en el menor tiempo posible. Luego de observar su comportamiento, nos damos cuenta que con ésta estructura utilizada (lista simple) se tiene un crecimiento lineal de accesos con respecto a los datos, lo cual se ve reflejado en la tabla. A mayor número de datos, mayor es el tiempo utilizado, teniendo en nuestro ultimo caso de prueba como resultado un número muy alto en la medida temporal, el cual no pudo tomarse en cuenta ya que era demasiado grande en comparación con los tiempos registrados en las primeras muestras.