Universidad de Buenos Aires
Facultad de Ingeniería
7506 - Organización de Datos

Cátedra Lic.Servetto

1er.Cuatrimestre-2013

Trabajo práctico

Índice

1. Pautas del Trabajo Práctico

Organización en grupos

El trabajo practico será realizado en grupos de 3 a 5 personas, dependiendo de la cantidad de estudiantes que haya en el cuatrimestre.

Fechas de Entrega

Las fechas de entregas son dispuestas por la cátedra en el Cronograma de la Materia y no tienen reentrega. Éstas tienen calificación, y su aprobación es indispensable para mantener la regularidad.

Aprobación

Para aprobar una entrega de Trabajo Práctico se debe cumplir con más del 80 % de los requisitos accesorios y con el 100 % de los concernientes a temas tratados en la Materia.

Entorno de trabajo y lenguaje

- Todos los trabajos prácticos serán evaluados en un sistema operativo Linux, en caso de no funcionar correctamente en dicho entorno se considerará reprobado.
- Los lenguajes de programación aceptados para los trabajos prácticos serán: C y C++

Entregables obligatorios

- Documentación de usuario y de diseño, donde se detallan la forma de uso del sistema y los problemas técnicos, con sus soluciones, respectivamente, desarrollando una introducción teórica a cada problema correspondiente a temas de la Materia y se argumentará la solución aplicada.
- Documentación de diseño.
- el código fuente de la aplicación con un Makefile para permitir la compilación completa de la solución

Cualquiera elemento anterior que falte, se considerará desaprobada la entrega y por ende el grupo recursará la materia.

2. Temática del Trabajo Práctico

Todo comenzó cuando los Salocin, un grupo de amigos de la facultad (estudiantes del Dpto. de Computación), cansados de tener que ir a internet a buscar la letra de una canción (para leer que dice y analizar su contenido), decidieron proponer un solución a este problema y realizar una aplicación OFFLINE. Pensaron en tener una "especie" de biblioteca de letras, en una aplicación, claro está que les de cierta potencialidad y funcionalidad,como por ej:

- Buscar letras de canciones por título.
- Buscar letras de canciones por autor.
- Buscar letras de canciones que contengan ciertas frases.
- Estadísticas de los patrones de búsqueda anteriores y que letras matchearon más veces.

Por otro lado, analizaron el costo de tener esta información plana, es decir, el costo de almacenamiento, y pensaron en que una vez indexadas e introducidas las letras en la "biblioteca", las mismas no deben almacenar el mismo tamaño. Es por eso que decidieron comprimir las letras luego de indexarlas y que en caso de que se necesite recuperar, se descomprima.

3. 1ra. entrega

En la presente entrega se deberá realizar la aplicación de la biblioteca que permita resolver lo siguiente:

1. Indexar letras, contenidas en archivos de texto plano, donde el formato del contenido del archivo es el siguiente:

   - 1ra línea: Autor-Año de grabación-título-idioma del tema.

   Cabe aclarar que el año de grabación es opcional.
   Por otro lado puede haber múltiples autores para una letra dada, con lo cual los mismos seran separados por el caracter punto y coma(;).
   El separador de información será el guión medio(-), como indica el ejemplo. Cualquier archivo que no cumpla con esto, DEBE ser descartado de la indexación.

2. Buscar letras de canciones por título.
3. Buscar leras de canciones por autor.
4. Buscar letras de canciones por frases.

3.0.1. Aclaraciones de la entrega

- La fase de indexación es única, es decir, no se debe realizar un demonio o servicio que está buscando agregar archivos a la biblioteca.
- El usuario debe tener una opción, para reindexar la biblioteca en el momento que desee.
- En caso de que exista una biblioteca al momento de la indexación, se le debe solicitar al usuario la opción de sobreescribir la indexación actual o realizarla en modo append, descartando las letras que ya fueron indexadas(Se descarta por título,Autor e idioma.
  Ej. Bob Marley-I shoot the sheriff en distintos años para un mismo idioma, estará una única vez en nuestra base.)
- Se le debe permitir al usuario cambiar el directorio de origen y destino de la indexación.
- La búsqueda por título y autor será exacta(Ej. sí se encuentra indexado únicamente Bob Marley como autor, y el usuario pide buscar Bob, no habrá resultados. Ej2: Sí se encuentra indexado el título Ambush in the Night, y se busca night, no devolverá resultados)
- La búsqueda por frase, debe cumplir el mismo estandar que el dado en las teóricas y contener la mayor cantidad de casos aplicables.
- Debe entregarse una aplicación intuitiva, con un menú sencillo para poder realizar cada operación.

3.0.2. Especificación técnica

A fin de orientar la implementación de las distintas estructuras de datos, se pide:

- En el diseño de la capa física, se deberá especificar las estructuras a utilizar para el manejo de espacios libres. Considerar el uso de mapa de bits para archivos en bloques, y listas encadenadas para registros de longitud variable.
- Se deberá implementar en la capa física, las rutinas y/o clases necesarias para trabajar con Archivos en Bloques y Archivos con registros de longitud variable. El diseño deberá especificar qué tipo de archivo se utilizará para cada estructura de datos.
- Los archivos indexados deberán ser guardados en su formato y contenido original dentro de un archivo de registros de longitud variable a fin de poder recuperarlos tal cual fueron ingresados a la aplicación.
- Se deberá implementar los árboles de búsqueda (búsquedas por autor) con un Árbol B+. El mismo deberá ser genérico, es decir, deberá soportar el uso de claves genéricas (pensar en una jerarquía de objetos, donde la clase padre define los métodos

necesarios para la comparación y ordenamiento de las clases herederas). El tamaño de los bloques, tanto para nodos internos como para nodos hoja, deberá ser fijo pero configurable en tiempo de compilación.

- Las claves genéricas, deberan contener sí o sí un identificador único generado en forma automática. A tal fin, se pueden utilizar distintos tipos de algoritmos para su generación, como ser: timestamp, autoincremental, entre otras.
- Arquitectura en capas. Se deberá diseñar la arquitetura del trabajo práctico definiendo claramente al menos tres capas: Física (manejo de archivos, bloques y estructuras básicas), Lógica (manejo de estructuras de indexación) y Aplicación (lógica propia de la solución planteada, que utilizará los servicios que le brinde la capa Lógica).
- El diseño deberá contemplar la ubicación física de los distintos archivos, incluyendo tanto a los de entrada y salida, como cualquier otro archivo intermedio (incluyendo la estrucutura de carpetas). Estas ubicaciones deberán ser configurables.
- Se deberán implementar el algoritmo de dispersión (hash) extensible para las búsquedas por términos exactos, definido anteriormente(implementar el índice de título). El tamaño de las cubetas (buckets) deberá ser fijo pero configurable al momento de la compilación.
- Para la busqueda por frases, se deberá implementar un índice de tipo invertido utilizando los conceptos vistos en el módulo de FTRS.
- Para poder hacer búsquedas por título y por autor más efectivas, se deberá incluir en el diseño normas de estandarización de dichos campos. (por ejemplo, para el caso del autor, apellido primero y nombre/s después, con una coma luego del apellido, o nombres primero y apellido al final, sin comas, y convención para el uso de mayúsculas -todo en mayúsculas-, o primeras letras).
- El formato y número de pantallas de la interfaz de usuario deberá ser definida también en el diseño. Sólo se pide que dicha interfaz quede bien documentada y con ejemplos en un manual de usuario de la aplicación.
- En la segunda entrega, se incluirá la compresión de las letras. Ver Especificación técnica de la Segunda Entrega.

3.0.3. Fecha de la 1ra. Entrega

Desde la fecha de presentación del TP, se dará un lapso de 7 (SIETE) semanas para presentar la entrega. Por lo tanto, la misma se realizará finalizando la semana 11 del cuatrimestre actual. Queda a disposición de cada grupo, si lo desea, hacer entregas parciales o realizar la entrega antes del lapso dado (RECOMENDABLE). La primer entrega NO TIENE posibilidad de reentrega.

4. 2da. entrega

En la presente entrega se deberá agregarle a la aplicación de la biblioteca desarrollada para la 1ra entrega, lo siguiente:

1. Compresor PPMC con exclusión, que permita comprimir las letras a medida que se indexan. El mismo debe permitir parametrizar el orden del algoritmo desde un archivo de configuración. El mismo debe permitir la descompresión, para poder recuperar la letra.
2. Módulo de estadísticas que debe arrojar los siguientes resultados:
   a) Búsquedas por títulos más buscadas.
   b) Búsquedas por autor más buscadas.
   c) Búsquedas por frase más buscadas.
   d) Tema que coincidió más veces en todas las búsquedas

   Estás estadísticas no son históricas, sino que a una corrida en particular del sistema.

4.0.4. Aclaraciones de la entrega

- La compresión se aplica sólo al momento de indexar.
- La descompresión se realiza cuando lo solicita el usuario. Para esto se le debe permitir en cada búsqueda o en un listado general, la opción de poder ver la letra de alguna canción.
- Cada tipo de estadística es independiente entre sí, salvo la última (Tema que más coincidio) que relaciona las 3 anteriores.

4.0.5. Especificación técnica

A fin de orientar la implementación de esta segunda parte, se pide:

- La implementación del algoritmo PPMC deberá contemplar el uso de una lista múltiple para los Órdenes con Árbol Genérico para los contenidos del Contexto. La implemataación del Compresor Aritmético deberá ser con enteros.
- Agregar a la implentación del algoritmo un informe con los Miss y Match de cada contexto del PPMC.
- La compresión deberá incluirse en las indexaciones de la primer entrega. Por lo tanto, el diseño inicial debe contemplar esta extensión.

- Sólo se comprimirán los registros de longitud variable que contienen los archivos originales (la compresión se realizará en forma independiente, es decir, cada archivo original se comprime independientemente del resto -no realizar una única compresión de la base de datos entera-).
- No se requiere una comparativa comparativa, simplemente en tiempo de ejecución.

4.0.6. Fecha de la 2da. Entrega

Desde la fecha de entrega de la primer parte (semana 11), se dará hasta el último día de la penúltima semana de clases (semana 15) para poder realizar la entrega. Queda a disposición de cada grupo, si lo desea, hacer entregas parciales o realizar la entrega antes del lapso dado (RECOMENDABLE). La segunda entrega SI TIENE la posibilidad de reentrega, en la semana siguiente a la devolución de la corrección de la entrega de la semana 15.

4.1. Aclaraciones generales de cada entrega

El grupo deberá entregar a cada tutor, un bosquejo que contenga el marco teórico/práctico de lo que se va a llevar a cabo para cada entrega, de manera que se pueda validar el camino a seguir por el grupo. De no cumplir alguna de las pautas generales o de faltar algún componente de los que se solicita en cada entrega, el tutor verá el porcentaje de avance y considerará si está aprobada/desaprobada la misma.

4.2. Tutores- Mails de contacto

Por cualquier duda, entrega o sugerencia, pueden enviar un mail a cada tutor de los que estén en la siguiente lista:

- Alejandro Ferrer - aleee.datos@gmail.com
- Maximiliano Stibel- mstibel.datos@gmail.com
- Nicolás Gallinal- nicoabie@gmail.com
- Nicolás Fernández Theillet- nflabo@gmail.com