Prácticas Recuperación de Información.
Grado en Ingeniería Informática.

P1. Ejercicio 1

Indexador para un sistema de desktop search

Estudie y pruebe el código
http://lucene.apache.org/core/6_3_0/demo/src-html/org/apache/lucene/demo/IndexFiles.html

De este ejercicio no debe entregar nada pero en la defensa de la práctica se le puede pedir que implemente variantes del mismo.

P1. Ejercicio 2

Buscador para un sistema de desktop search

Estudie y pruebe el código
http://lucene.apache.org/core/6_3_0/demo/src-html/org/apache/lucene/demo/SearchFiles.html

De este ejercicio no debe entregar nada pero en la defensa de la práctica se le puede pedir que implemente variantes del mismo.

P1. Ejercicio 3.

Descargar e indexar la colección Reuters 21578
http://kdd.ics.uci.edu//databases/reuters21578/reuters21578.html

Tener en cuenta:

-Los archivos a indexar son los *.sgm
-El tag REUTERS identifica el principio y final de cada artículo.
-Debéis **indexar los campos TOPICS, TITLE, DATELINE, BODY y DATE** que aparecen en los documentos Reuters. Además debéis indexar los campos PathSgm con la ruta del archivo .sgm donde se encuentra el documento, **OldId** y **NewId** con los valores de esos campos que se encuentran en el campo TOPICS del documento REUTERS, **Hostname** y **Thread** que identifican el host y thread que se encargaron de la indexación de este documento.
-Cada tópico individual en TOPICS está identificado por un tag D.
-Si el BODY finaliza con "Reuter\n&#3;" ó "REUTER\n&#3;" (boiler plate) el parser debe eliminar estos textos para no indexarlos.

Se os **proporciona un parser** de la colección Reuters que ya se ocupa de la extracción de los campos y del boiler plate. El parser debe estar desacoplado del resto del código y tocarlo sólo si es imprescindible. De esta forma el código de indexación será fácilmente adaptable para indexar otra colección, lo que se hará en otra práctica.

Los archivos a indexar están dispuestos en una sóla carpeta y un nivel pero el indexador debe funcionar aunque estuviesen dispuestos en cualquier carpeta en el árbol de directorios a partir de la carpeta raíz contenedora. Se supondrá además que cualquier nombre de archivo reut2-xxx.sgm con x de 0 a 9 es válido. En distintas carpetas pudiera haber nombres *.sgm iguales pero al estar en carpetas distintas no se puede suponer que tengan el mismo contenido. Por tanto es necesario hacer

pruebas del indexador disponiendo los archivos .sgm en distintas carpetas y niveles del sistema de ficheros.

Indexación del campo DATE:

La clase SimpleDateFormat de Java permite parsear y formatear fechas. En nuestro caso basta usar el constructor de la clase pasándole un string con el formato de la fecha según el formato del campo DATE de la colección Reuters
Invocando el método parse del objeto creado en el paso anterior sobre el string obtenido del campo DATE se obtiene un objeto de la clase Date de Java
Con el método dateToString de la clase DateTools de Lucene se convierte el objeto Date de Java en un string para almacenar en el campo correspondiente del índice.
Todo este proceso es necesario para que posteriormente las fechas puedan ser reconocidas por Lucene, en particular en queries sobre rangos de fechas. Puede verse la sintaxis de la queries en:

http://lucene.apache.org/core/6_3_0/queryparser/org/apache/lucene/queryparser/classic/package-summary.html#package_description

La práctica tiene una serie de opciones de indexación, que se especifican con estos argumentos

-openmode openmode (el open mode será append, create, o create_or_append)
-index pathname (ruta de la carpeta que contiene o contendrá el índice)
-coll pathname (ruta de la carpeta raíz que contiene los archivos *.sgm de la colección)
-multithread (opción para la creación concurrente de índices)
-addindexes (opción para la creación de varios índices y su posterior fusión)
Si la opción -multihread no está habilitada, el único hilo de ejecución indexará todos los archivos *.sgm que se encuentran en la carperta raiz indicada en la opción -coll y en todo el árbol de carpetas que cuelgue de ella. Si la opción -multihread está habilitada se creará un hilo por cada subcarpeta de primer nivel de la carpeta pathname especificada en la opción -coll. Cada hilo creará un índice de forma concurrente con todos los archivos *.sgm que cuelga de esa subcarpeta que a su vez puede contener subcarpetas hasta cualquier nivel. El número de hilos vienen dado sólo por el número de subcarpetas de primer nivel y se puede suponer que en la carpeta raíz no hay archivos *.sgm y que se usa sólo como contenedera de las subcarpetas que contienen los archivos *.sgm. Si además la opción -addindexes está habilitada, cada hilo creará un índice en una subcarpeta de primer nivel de la carpeta -index pathname con un nombre que recoja la identificación de la subcarpeta de primer nivel que contiene los archivos con los que se creó el índice. Finalmente con el método addIndexes de IndexWriter class se fusionan los distintos índices y se coloca el índice fusionado en una subcarpeta del mismo nivel que los otros índices y con un nombre apropiado.
Si la opción -multihread está habilitada pero la opción -addindexes no está habilitada, se construye un índice de forma concurrente para los archivos correspondientes a las carpetas indicadas en la opción -colls, pero no se pueden construir índices parciales, sino que todos los threads escriben concurrentemente el índice final. Debe compararse la eficiencia de -multithread con respecto a un sólo hilo, y de -multithread con -addindexes o sin -addindexes, además de comprobar que producen los mismos índices, i.e. índices que contienen los mismos documentos y términos.

En general para estas y todas las funcionalidades de la práctica se valorará la calidad y eficiencia de las soluciones.

Se proporciona también un **ejemplo de un pool de threads** que se puede adaptar para esta práctica.

La práctica tiene también una serie de opciones de procesado del índice construido, que se

especifican con estos argumentos:

-indexin indexfile (indexfile es la ruta donde está el índice que se procesa)
-best_idfterms field n (devuelve ordenados por idf, con el número de orden y el valor de idf, los n mejores términos del campo field en el índice indicado por -indexin).
La frecuencia de documento de un término en la colección df_t es el número de documentos en los que aparece el término, idf mide la especificidad del termino idf = log (N/df_t)), donde N es el número de documentos de la colección.
-tfpos field term
Dado un campo y un término construye un listado con la siguiente información: docid de Lucene, PathSgm, OldId y NewId del documento donde se encuentra el término; tf (frecuencia del término en el documento), posiciones del término en el documento, df del término. El campo debe haberse creado con las opciones de indexación de docs, freqs y positions.
-termstfpos1 docid field ord
Dado un docid de Lucene y un campo, construye un listado con la siguiente información: término, docid de Lucene, PathSgm, OldId y NewId del documento donde se encuentra el término; tf (frecuencia del término en el documento), posiciones del término en el documento, df del término. El campo debe haberse creado con las opciones de indexación de docs, freqs y positions. El listado vendrá ordenado según el valor de ord: 0 alfabético, 1 por orden decreciente de tf, 2 por orden decreciente de df.
-termstfpos2 PathSgm NewID field ord (lo mismo que la opción -termstfpos pero para un documento identificado por su PathSgm y NewID).

La práctica tiene algunas opciones de procesado de un índice y a partir de este procesado construcción de un nuevo índice:
-indexin indexfile (indexfile es la ruta donde está el índice que se procesa)
-indexout indexfile (indexfile es la ruta donde se crea el índice construido con las opciones de procesado)
-deldocsterm field term (borra los documentos que contienen el término especificado)
-deldocsquery "query" (borra los documentos que satisfacen la query. Esta opción y la anterior pueden modificar el índice especificado en -indexin y no es necesario especificar -indexout)
-summaries (crea un índice con resúmenes de los documentos, en esta opción es necesario especificar las rutas para -indexin y -indexout)
-multithread n (creación de resúmenes con n hilos)
En la opción -summaries se lee el índice indicado en -indexin y se crea otro indice indicado en -indexout que contiene los mismo campos y contenidos y un campo nuevo Resumen que contiene las dos frases más similares del campo body con respecto al campo título para cada documento. Para ello basta crear un índice en memoria para cada documento y lanzar una query con el título frente al índice formado por las frases del body. Si el título es nulo, el resumen contiene las dos primeras frases del body. Si el body es nulo, el resumen es nulo. Si el body sólo tiene una frase el resumen contendrá sólo esa frase. Si se indica -multithread, el procesamiento de los resúmenes se hará con n hilos y cada hilo se ocupa de numDocs/n documentos del índice donde numDocs es el número de documentos del índice.
Las opciones de creación y procesado de índice son mutuamente exclusivas. Primero se crea el índice y en otra ejecución se procesa. **También se puede suponer que las opciones de procesado de borrado de documentos y de creación del índice con resúmenes son exclusivas.**

Entrega P1

-LAS PRÁCTICAS SON POR PAREJAS.
-EN EL CASO DE COPIA DE PRÁCTICAS, TODOS LOS ALUMNOS IMPLICADOS

PERDERÁN LA NOTA TOTAL DE PRÁCTICAS.
-Se sube al repositorio SVN antes de la fecha límite indicada
-Se crea una carpeta **P1** y se sube a esa carpeta un proyecto Java con nombre **jri-indexer** o un proyecto Maven con nombre **mri-indexer** (artifactId). SOLAMENTE puede subirse ese proyecto y en esa carpeta. SOLO UNO DE LOS ALUMNOS de la pareja de prácticas puede subir la práctica, si la suben los dos, los scripts de bajada y procesamiento detectarán copia y se invalidará la práctica. Si la subida no se hace en las condiciones (nombre de carpeta y proyecto) establecidas, los scripts de procesamiento tampoco detectarán la práctica como correcta y **no se evaluará.**
-Además de la entrega habrá una revisión in situ de las prácticas donde se pedirán cambios que deberán implementarse en el aula de prácticas para cualquier parte de las mismas. Cualquier miembro de la pareja de prácticas debe responder a cualquier aspecto de la defensa, y también se pedirá que cada uno implemente una variante distinta de la práctica. Por tanto aunque el trabajo es en equipo cada miembro debe conocer al detalle lo realizado por el compañero.
-Si se detecta **copia en prácticas** se aplicará lo establecido en las normas de la asignatura.
-Al principio de la defensa también se pedirá que se baje el proyecto subido al SVN en la fecha límite y que las pruebas se hagan sobre esa versión para lo que debéis probar antes los comandos correspondientes de SVN, por ejemplo:

svn checkout https://svn.fic.udc.es/grao3/ri/17-18/login/P1 -r {'aaaa-mm-dd hh:mm'}

donde la fecha corresponde a las 00:01 del día siguiente al límite de entrega. Y después de bajar el proyecto del repositorio, con Eclipse Importar el proyecto Java o Maven.