Trabajo MiniTREC: Instrucciones para la entrega del sistema de recuperación tradicional

Escuela de Ingeniería y Arquitectura
Depto. de Informática e Ingeniería de Sistemas

1. Descripción general de la tarea

Siguiendo las pautas establecidas en las tres primeras prácticas para la utilización de Lucene, se propondrá un sistema de recuperación que realice la indexación de la colección facilitada y parsee las necesidades de información.

Para comenzar a realizar esta tarea se cuenta con:

- La colección de metadatos Dublin Core del repositorio Zaguan facilitada por los profesores en formato XML. El fichero comprimido «`recordsdc.zip`» contiene dicha colección de documentos.
- Las necesidades de información elegidas entre las propuestas por todos los equipos. El fichero «`necesidadesInformacionElegidas.xml`» contiene las **propuestas elegidas**. El formato de este fichero XML es el especificado en la tarea de entrega de necesidades de información. A título informativo, también se puede descargar el fichero «`necesidadesInformacionEnviadas.xml`» que contiene el conjunto de todas las necesidades de información enviadas por los alumnos.

Para la indexación cada equipo elegirá el esquema de indexación y las técnicas (tokenización, normalización, ...) que considere más adecuadas.

Igualmente, en las búsquedas cada equipo puede utilizar las técnicas que estime más adecuadas. El sistema construido recibirá como entrada las necesidades de información, y las procesará y transformará de la forma que considere más oportuna para construir la consultas que se resolverán con Lucene. **Nota:** No se debe limitar el número de

resultados (documentos) devueltos por Lucene. Lo realmente importante es la ordenación por relevancia de estos resultados.

Aunque el sistema se debe ajustar en base a las características que se incluyen en las necesidades de información elegidas, los programas de indexación y búsqueda deben ser suficientemente flexibles para aceptar como entrada otras necesidades de información con características similares.

2. Requisitos técnicos de los programas a desarrollar

Asumiendo que se ha elegido *IndexFiles* como nombre de la clase que contiene el programa de indexación, este programa se debería poder ejecutar desde una terminal (o línea de comandos) con la siguiente sintaxis para los 2 argumentos de entrada que debe recibir de forma obligatoria:

```
java IndexFiles -index <indexPath> -docs <docsPath>
```

- El argumento *-index <indexPath>* permite indicar la ruta del directorio donde se generarán los índices de Lucene.
- El argumento *-docs <docsPath>* permite indicar la ruta del directorio que contiene la colección de documentos a indexar.

Respecto al programa de búsqueda, asumiendo que se ha elegido *SearchFiles* como nombre de la clase que contiene el programa de búsqueda, este programa se debería poder ejecutar desde una terminal (o línea de comandos) con la siguiente sintaxis para los 3 argumentos de entrada que debe recibir de forma obligatoria:

```
java SearchFiles -index <indexPath> -infoNeeds <infoNeedsFile> -output <resultsFile>
```

- El argumento *-index <indexPath>* permite indicar la ruta del directorio donde se almacenan los índices de Lucene.
- El argumento *-infoNeeds <infoNeedsFile>* permite indicar la ruta del fichero que contiene las necesidades de información. Se debe utilizar como formato de ese fichero el especificado en la tarea de entrega de necesidades de informacion.
- El argumento *-output <resultsFile>* permite indicar la ruta del fichero de texto donde se generarán los resultados del sistema para las necesidades de información. Cada línea del fichero contiene el identificador de la necesidad de información, un *tabulador*, y el identificador de uno de los documentos de la colección recuperados para esa necesidad de información. Como identificador del documento se utilizará el identificador del fichero Dublin Core. A continuación se muestra un ejemplo:

```
102-5    oai_zaguan.unizar.es_5460.xml
102-5    oai_zaguan.unizar.es_6453.xml
...
207-4    oai_zaguan.unizar.es_10751.xml
207-4    oai_zaguan.unizar.es_7131.xml
```

Para cada necesidad de información el fichero de resultados deberá contener el ranking ordenado de documentos recuperados. Este es un requisito imprescindible para poder realizar los juicios de relevancia, y la evaluación de los sistemas de cada equipo. **Nota:** Para facilitar el procesamiento posterior de este fichero de resultados en la evaluación de los sistemas, no se debe incluir ningún tipo de cabecera o separador al incluir los resultados de una nueva consulta.

Para los programas de indexación y búsqueda **se exigirá la definición y utilización de analizadores personalizados, al menos uno,** siguiendo las pautas recomendadas en el anexo A.1.2 de la práctica 1. Se recomienda revisar también el apartado 2.5 de la práctica 1 donde también se solicitó la definición de un analizador personalizado.

Para el procesamiento de las necesidades de información y su transformación en las consultas que recibirá el motor de búsqueda de Lucene, también puede ser de interés considerar la utilización de herramientas de procesamiento del lenguaje natural (NLP) que faciliten el etiquetado gramatical o el reconocimiento de entidades con el objeto de crear subconsultas sobre campos específicos. Por ejemplo, OpenNLP[1] es una librería Java para NLP que ofrece en su API herramientas de etiquetado gramatical[2] o reconocimiento de entidades.[3] Hay modelos ya entrenados para estas herramientas en distintos idiomas.[4]

3. Resultados a entregar

Como resultado de esta tarea se subirán a Moodle los siguientes entregables:

- Un pequeño de informe en formato PDF, cuya longitud estará entre 2 y 5 páginas, donde se describa brevemente:
  - La arquitectura del software desarrollado: al menos un pequeño diagrama de clases acompañado de un párrafo describiendo dicho diagrama.
  - Un breve resumen razonado de:

[1] `https://opennlp.apache.org/`

[2] `https://opennlp.apache.org/docs/1.9.3/manual/opennlp.html#tools.postagger.tagging.api`

[3] `https://opennlp.apache.org/docs/1.9.3/manual/opennlp.html#tools.namefind.recognition.api`

[4] Ver `http://opennlp.sourceforge.net/models-1.5/` y `https://cavorite.com/labs/nlp/opennlp-models-es/`.

  - las técnicas utilizadas en el proceso de indexación (descripción de analizadores utilizados) y los índices (campos) que se han creado
  - ; las técnicas utilizadas en el proceso de parseo de consultas (descripción de analizadores utilizados o cualquier otro tipo de procesamiento)
  - ; y el algoritmo elegido para el cálculo del ranking.[5]
- Un breve comentario sobre los resultados obtenidos sobre la colección de Zaguan para las necesidades de información contenidas en el fichero «`necesidadesInformacionElegidas.xml`».

- Un fichero de texto denominado *equipo<NNN>.txt*[6] con los resultados obtenidos para las necesidades de información contenidas en el fichero «`necesidadesInformacionElegidas.xml`».
- Un fichero comprimido con el código fuente debidamente documentado y opcionalmente cualquier otro tipo de documentación que resulte de interés para el sistema desarrollado.

Solo uno de los alumnos de cada equipo realizará la entrega a través de esta tarea.

4. Seguimiento del trabajo

Se recuerda que se debe concertar al menos una sesión de tutorías con el profesor Javier Nogueras con anterioridad a la entrega del sistema de recuperación tradicional. Para solicitar una hora concreta, hay que iniciar sesión en Google con una cuenta Google, ir al enlace de reserva de citas para reuniones TP6 del calendario Google de la asignatura y seleccionar una tutoría disponible, indicando en el apartado «Descripción» los nombres de los integrantes del grupo que solicitan la tutoría. Las tutorías deben reservarse con al menos 24 horas de antelación sobre la hora prevista de la tutoría. Al reservar las tutorías, comprobad que la zona horaria que tenéis configurada en vuestra cuenta Google es la correspondiente a Madrid («(GMT+2) Madrid» cuando estemos en horario de verano, y «(GMT+1) Madrid» con horario de invierno. En la interfaz web, la zona horaria que se está utilizando aparece reflejada en la cabecera de la columna que muestra las horas de los evento).

[5] A través del método *setSimilarity* de la clase *IndexSearcher* se puede cambiar el modelo a utilizar. Por defecto se usa la clase *BM25Similarity* basada en el modelo probabilístico BM25. Si se quiere utilizar el modelo vectorial, la clase *org.apache.lucene.search.similarities.ClassicSimilarity* proporciona una implementación.

[6] *<NNN>* se corresponde con el número del equipo utilizando dos dígitos.