Universidad Autónoma de Madrid, Escuela Politécnica Superior

Grado en Ingeniería Informática

Búsqueda y Minería de Información 2013-2014

Práctica 4: Funciones de ranking avanzadas: búsqueda proximal, PageRank y combinación de rankings

Fechas

- Comienzo: 14 de marzo
- Entrega: 4 de abril (hora límite 12:00)

Objetivos

El objetivo de esta cuarta práctica es desarrollar modelos de búsqueda y funciones de ranking adicionales a los que tenía el motor de búsqueda de la práctica 3, es decir, los modelos de búsqueda Booleano, vectorial y literal. En concreto, se van a desarrollar:

- Búsqueda proximal
- Ranking basado en grafos: PageRank
- Combinación de rankings

Para ello se extenderá la jerarquía de paquetes y clases Java de la práctica 3.

Colección de documentos

Los índices y buscadores de esta práctica se van a construir y evaluar sobre las tres colecciones de documentos HTML de la práctica 2: **clueweb-1K.tgz** (10 MB), **clueweb-10K.tgz** (82 MB) y **clueweb-100K.tgz** (865 MB).

Se recuerda que cada colección está formada por:

- Un fichero comprimido **docs.zip** con los documentos HTML.
- Un fichero **queries.txt** con las consultas de búsqueda, en el formato `consulta_id:consulta`.
- Un fichero **relevance.txt** con aquellos documentos relevantes a cada consulta. Por cada consulta, se da la lista de documentos siguiendo el formato `consulta_id \t documento_1 \t documento_2 \t ... documento_N`.

Adicionalmente se proporciona:

- Un fichero **links.zip**, **l** con los hiperenlaces entre los documentos HTML de las colecciones 1K, 10K y 100K. Los grafos de hiperenlaces se utilizarán para la evaluación del algoritmo PageRank, a desarrollar en esta práctica.

  El formato del fichero de hiperenlaces es el siguiente

  <id_cluweb> <n outlinks > [id_clueweb_link]*

  Por ejemplo, la siguiente línea:

  clueweb09-en0006-91-33158 178 clueweb09-en0008-09-17290 clueweb09-en0008-09-17034 clueweb09-en0008-09-17002

  Indica que el documento clueweb09-en0006-91-33158 tienen un total de 178 enlaces a otros documentos. Entre estos enlaces hay 3 enlaces a documentos que están en esa misma colección que corresponden a los documentos clueweb09-en0008-09-17290, clueweb09-en0008-09-17034 y clueweb09-en0008-09-17002.

Ejercicios

Ejercicio 1: Búsqueda proximal [4 puntos]

Se pide desarrollar una clase **es.uam.eps.bmi.search.searching.ProximalSearcher** que implemente **es.uam.eps.bmi.search.searching.Searcher**, y que a través del método **search** realice búsquedas proximales sobre un índice dado.

La búsqueda proximal puntúa los resultados según la cercanía con la que aparecen las palabras de la consulta en los documentos. La función de *scoring* representa una alternativa a otras funciones de ranking como la vectorial o la Booleana.

La clase deberá tener un método **main** que solicite al usuario y ejecute de forma iterativa consultas sobre un índice de la colección de documentos de la práctica.

Así mismo, análogamente a la práctica 3, el buscador se ejecutará y evaluará con las consultas y juicios de relevancia dados en los ficheros de cada una de las 3 colecciones ClueWeb. Se pide calcular y reportar los valores promedios de P@5 y P@10 obtenidos para las consultas de cada colección.

La calificación de este ejercicio se repartirá como sigue:

- Clase ProximalSearcher [3 puntos]
- Un documento **bmi1314_XXXX_p4_YY_proximal.pdf** con un análisis de resultados que incluya al menos una tabla visualizando los valores de P@5 y P@10 promedios del motor de búsqueda proximal para las 3 colecciones ClueWeb, y ejemplos explicativos de documentos recuperados para varias consultas. Se deberá entregar el código desarrollado para el cálculo de los valores de precisión [1 punto]

Ejercicio 2: PageRank [3 puntos]

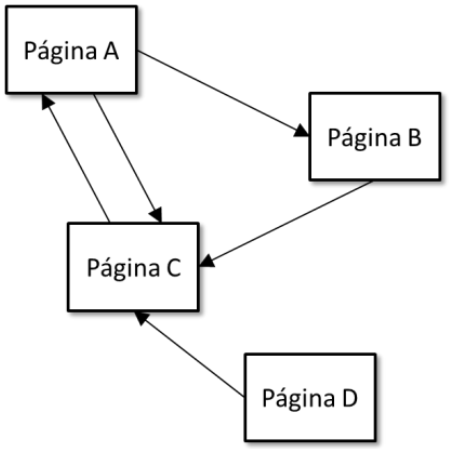
Se pide desarrollar una clase **es.uam.eps.bmi.search.ranking.graph.PageRank** que al menos tenga el siguiente método:

- **double getScoreOf(String documentId)**, que devolverá el valor de PageRank asociado al documento del identificador dado.

Aparte del método anterior se recomienda que la clase tenga un método para calcular los valores de PageRank de los documentos a partir de un fichero de enlaces dado. También se recomienda que la clase guarde en disco (p.e. asociando al índice) los valores de PageRank calculados, con el fin de que se pre-calculen *offline* (en tiempo de indexado) y estén disponibles en el momento de procesar consultas.

Para probar y validar la implementación de PageRank realizada, se pide desarrollar dos programas (métodos **main**) en las siguientes clases:

- **es.uam.eps.bmi.search.ranking.graph.PageRankTest1**, que ejecutará PageRank sobre el grafo dado en la figura de abajo, y que irá visualizando por pantalla el estado del vector de PageRank en cada una de sus iteraciones.



- **es.uam.eps.bmi.search.ranking.graph.PageRankTest2**, que ejecutará PageRank sobre el grafo asociado a un fichero de links de documentos, y que visualizará por pantalla los nombres de los 10 documentos con mayor valor de PageRank. Se entregará un fichero **bmi1314_XXXX_p4_YY_pagerank.txt** con la salida de la ejecución.

Indicaciones:

- Se recomienda, al menos inicialmente, llevar a cabo una implementación con la que los valores de PageRank sumen 1, para ayudar a la validación de la misma. Posteriormente, si se desea, se pueden escalar (o no, a criterio del estudiante) los cálculos omitiendo la división por el número de páginas.
- Será necesario tratar los nodos sumidero tal como se ha explicado en las clases de teoría. Se recomienda comprobar el correcto funcionamiento del algoritmo en un pequeño grafo con algún nodo sumidero.

La calificación de este ejercicio se repartirá como sigue:

- Clase PageRank [1.5 puntos]
- Visualización de la matriz de valores PageRank sobre el grafo de arriba [0.5 puntos]
- Ejecución de PageRank sobre la colección de documentos de la práctica, incluyendo la lectura y procesado de los ficheros HTML [1 punto]

Ejercicio 3: Combinación de rankings [3 puntos]

Se pide desarrollar la clase **es.uam.eps.bmi.search.ranking.aggregation.WeightedSumRankAggregator** que realice agregación de rankings de documentos mediante suma ponderada de sus *scores*.

Se deja libertad para el diseño de la clase **WeightedSumRankAggregator**, pero se exige que la clase proporcione un método para generar un ranking de scores, **List<ScoredTextDocument>**, a partir de otros rankings de entrada.

Para probar y validar la implementación de la clase se pide realizar un método **main** que permita ejecutar consultas con buscadores implementados en las prácticas 3 y 4, y que compare los rankings obtenidos con agregaciones de los mismos. En concreto, se pide se comparen los siguientes rankings:

- TFIDFSearcher
- ExactMatchingSearcher
- ProximalSearcher
- TFIDFSearcher + PageRank
- ExactMatchingSearcher + PageRank
- TFIDFSearcher + ProximalSearcher + PageRank

La comparativa ha de ser cualitativa (p.e. analizando qué documentos comunes y no comunes aparecen en los top *k* resultados de los rankings para una consulta) y cuantitativa (p.e. calculando métricas como P@*k*).

El resultado de este ejercicio constará de:

- La clase **WeightedSumRankAggregator**, incluyendo el método **main** para prueba de agregaciones de rankings [1 punto]
- Un fichero **bmi1314_XXXX_p4_YY_rank_aggregation.pdf** con el análisis de resultados de la ejecución del método **main** [2 puntos]

Ejercicio 4 opcional: Crawler [2 puntos extra]

Se pide desarrollar una clase **es.uam.eps.bmi.crawling.GenericCrawler** que realice el proceso de crawling de un sitio Web dado (a través de una URL de partida). El crawler a desarrollar debería permitir su configuración, p.e. para establecer los dominio(s) de descarga admitidos, el número máximo de páginas Web a descargar, la carpeta en la que almacenar las páginas Web descargadas, etc.

Se deja libertad para el diseño del crawler; tan sólo se exige que la clase **GenericCrawler** tenga un método **main** que permita ejecutar el crawler (estableciendo la configuración necesaria), y que una vez haya concluido tal ejecución, indexe las páginas Web descargadas, genere el grafo de hiperenlaces correspondiente, y calcule los valores de PageRank de esas páginas, mostrando finalmente las 10 páginas con mayor valor de PageRank.

Se permite y se recomienda que las pruebas del crawler se restrinjan a un único dominio (acotado).

Se probará el índice construido mediante la ejecución de consultas de prueba (que elegirá el propio estudiante) utilizando los modelos de ránking ya desarrollados en ejercicios anteriores.

Calificación

Esta práctica se calificará con una puntuación de 0 a 10 (+ 2 puntos extra para la parte opcional) atendiendo a las puntuaciones individuales de ejercicios y apartados dadas en el enunciado. El peso de la calificación de la práctica en la calificación final de prácticas es del **25%**.

Entrega

La entrega de esta práctica consistirá en un fichero ZIP con el nombre **bmi1314_XXXX_p4_YY.zip**, donde XXXX debe sustituirse por el grupo 2461 ó 2462 según corresponda, e YY debe sustituirse por el número de pareja (01, 02, …, 10, …). Este fichero contendrá:

- Una carpeta **src** con todos los paquetes y ficheros fuente .java desarrollados.
- El documento **bmi1314_XXXX_p4_YY_proximal.pdf** del ejercicio 1.
- El documento **bmi1314_XXXX_p4_YY_pagerank.txt** del ejercicio 2.
- El documento **bmi1314_XXXX_p4_YY_rank_aggregation.pdf** del ejercicio 3.

Dicho fichero se enviará por el enlace habilitado al efecto en el curso **Moodle** de la asignatura.