Estructuras de Datos

Fecha entrega: 20/05/2021

Bibliografía sugerida: MIR [1] Capítulos 8, Croft [2] Capítulo 5, MAN [3] Capítulos 4.

1. Codifique un script que procese documentos de un directorio y arme los índices que permitan soportar búsquedas booleanas. Grafique la distribución de tamaños de las *posting lists*. Calcule el *overhead* de su índice respecto de la colección. Calcule el overhead para cada documento. ¿Qué conclusiones se pueden extraer? Codifique un segundo script que permita mostrar la posting list para un término dado. En este caso, su script solamente debe mostrar la lista de DocIDs (ordenada), de la siguiente manera:
   ```
   192
   234
   556
   ...
   ```

2. Codifique un script que empleando la estrategia TAAT sobre el índice creado en el ejercicio 1 y operaciones sobre conjuntos permita buscar por dos o tres términos utilizando los operadores AND, OR y NOT.

3. Utilizando el código anterior ejecute corridas con la colección Wiki-Small[1] y el siguiente subset de queries[2] y mida el tiempo de ejecución en cada caso. Para ello, utilice los siguientes patrones booleanos:

   - Queries $|q| = 2$
     - t_1 AND t_2
     - t_1 OR t_2
     - t_1 NOT t_2
   - Queries $|q| = 3$
     - t_1 AND t_2 AND t_3
     - (t_1 OR t_2) NOT t_3
     - (t_1 AND t_2) OR t_3

   ¿Puede relacionar los tiempos de ejecución con los tamaños de las listas? (pruebe con el índice en disco o cargándolo completamente en memoria antes). ¿Qué conclusiones se pueden extraer?

4. Codifique un script que indexe una colección que requiera el volcado parcial a disco (asumiendo que existe un límite de memoria). Su script debe recibir un parámetro n que indica cada cuántos documentos se debe hacer el volcado a disco. Al finalizar, debe unir (*merge*) los índices parciales. Para las pruebas use la colección Wiki-Small y varios valores de n (por ejemplo, $n = 10\,\%$ del tamaño de la colección). Registre los tiempos de indexación y de *merge* por separado. Presente sus resultados en gráficos adecuados. Agregue un segundo script que permita mostrar la postings list para un término dado.

5. Modifique el script del ejercicio 1 para armar un archivo invertido con información de frecuencias. Luego, implemente consultas utilizando el modelo vectorial utilizando tres esquemas de ponderación y/o ranking diferentes. El script debe retornar una lista ordenada por score de pares $< DocID, score >$, con el siguiente formato:
   ```
   123 0.5
   2 0.46
   456 039
   ...
   ```

[1] `http://dg3rtljvitrle.cloudfront.net/wiki-small.tar.gz`
[2] `http://www.labredes.unlu.edu.ar/sites/www.labredes.unlu.edu.ar/files/site/data/ri/queries_2y3t.txt`

6. Modifique el script del ejercicio 1 para armar un archivo invertido posicional a nivel de palabra. Luego, implemente consultas con operadores de proximidad y búsquedas booleanas por frases. El formato de salida debe ser el mismo que el del ejercicio anterior.

7. Agregue *skip lists* a su índice del ejercicio 1 y ejecute un conjunto de consultas AND sobre el índice original y luego usando los punteros. Compare los tiempos de ejecución con los del ejercicio 2. Luego, agregue un script que permita recuperar las skips list para un término dado. En este caso la salida deberá ser la lista ordenada por DocId.

8. A partir de un conjunto de *posting lists* provistas[3] realice un programa que arme el vocabulario utilizando un B+Tree por un lado y un archivo binario con la información de las postings lists (DocId y frecuencias) por el otro. Use DGaps y agregue *skip-lists*. Ademas, agregue un script que permita recuperar: la lista de DGaps, las skip-lists y la postings list para un término dado (en todos los casos la salida deberá ser una lista de DocIds ordenada).

9. A partir del archivo de palabras en inglés (words-en.txt) calcule el tamaño necesario para almacenarlo en memoria sin comprimir o como dictionary-as-string. Haga una notebook y calcule la distribución de longitudes de las palabras y las estadísticas básicas. Conviene tomar una valor máximo de palabra? Justifique.

10. Sobre la colección Dump10k escriba un programa que realice una evaluación TAAT y otro usando DATT. Compare los tiempos de ejecución para un conjunto de *queries* dados[4]. Separe su análisis por longitud de *queries* y de *posting lists*.

11. A partir de la colección Wiki-Small construya el índice invertido con información de frecuencias y comprímalo utilizando Elias-Gamma y Variable-Lenght Codes. Calcule tiempos de compresión/descompresión y tamaño resultante en cada caso. Realice dos experimentos, uno codificando con DGaps y otro sin codificar. Compare los tamaños de los índices resultantes.

Referencias

[1] Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval: The Concepts and Technology Behind Search*. Addison-Wesley Publishing Company, USA, 2nd edition, 2008.

[2] Bruce Croft, Donald Metzler, and Trevor Strohman. *Search Engines: Information Retrieval in Practice*. Addison-Wesley Publishing Company, USA, 1st edition, 2009.

[3] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[3] `http://www.tyr.unlu.edu.ar/tallerIR/2014/data/dump10k.tar.gz`
[4] `http://www.labredes.unlu.edu.ar/sites/www.labredes.unlu.edu.ar/files/site/data/ri/queriesDump10K.txt.tar.gz`