TP – Estructuras de Datos

Universidad Nacional de Lujan

Materia: Recuperación de la Información

Profesores: Gabriel Tolosa
Pablo Lavallen

Alumno: Leonardo Duville

Estructuras de Datos

Fecha entrega: 15/05/2024

INFORMACION IMPORTANTE: En el punto 1 genero 3 archivos:

- dict_terms: Diccionario con id de términos para que coincidan IDs en el índice final.
- dict_files: Diccionario con el id de documentos de la colección.
- Vocabulary: Diccionario donde guardamos como clave el id de cada termino y en el valor la tupla (seek, df).

Estos puntos se generarán en el punto 1 y hay que pegarlos en las carpetas de cada punto que se vaya a probar para que funcione correctamente.

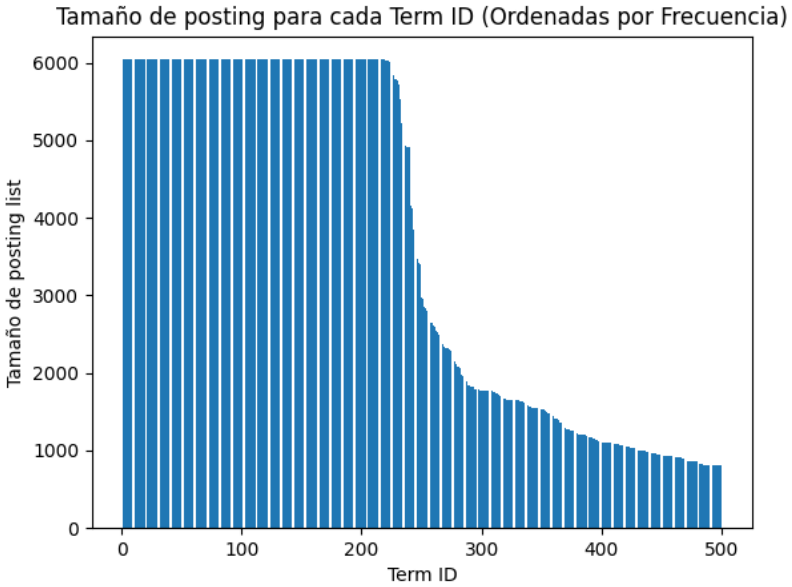
1) Codifique un script que indexe una colección[1] que requiera el volcado parcial a disco (asumiendo que existe un límite de memoria). Su script debe recibir un parámetro *n* que indica cada cuántos documentos se debe hacer el volcado a disco. Al finalizar, debe unir (merge) los índices parciales. Para las pruebas use la colección snapshot de Wikipedia[2] y varios valores de *n* (por ejemplo, n = 10% del tamaño de la colección). Registre los tiempos de indexación y de merge por separado. 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?

   En la tabla podemos ver los tiempos de indexación y de merge de la colección Wiki Small. Los mismos son similares y se ve una tendencia a aumentar a medida que crece n.

WIKI SMALL		
Limite Documentos	Time Indexing (sec)	Time Merge (sec)
n = 604. (10%)	89.10	6.09
n = 1812. (30%)	91.45	6.14
n = 3021. (50%)	89.67	6.10
n = 6043. (100%)	98.77	6.57

En el siguiente grafico se observa el tamaño de las posting lists de los primeros 500 términos del vocabulario. Los mismos se encuentran ordenados por DF de mayor a menor, teniendo los términos con mayor tamaño de posting list primero. Con esto podemos ver que hay pocos términos con un DF alto y muchos términos con DF bajo.

[1] Almacene docID o docID+frecuencia de acuerdo a un parámetro de entrada.
[2] http://dg3rtljvitrle.cloudfront.net/wiki-small.tar.gz(debug), http://dg3rtljvitrle.cloudfront.net/wiki-large.tar.gz (run)



En las siguientes tablas se llega a apreciar una diferencia notable el tamaño del índice respecto de la colección.

WIKI SMALL		
Collection Size	Index	Overhead
157 MB	34,5 MB	122,5 MB

Acá se calculó un promedio dividiendo los tamaños de la colección y del índice por la cantidad de documentos para calcular un overhead de cada documento.

WIKI SMALL		
Average Document Size (Collection)	Average Document Size (Index)	Overhead
157 MB	34,5 MB	0,020 MB

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

3) Utilizando el código e índice anteriores ejecute corridas con el siguiente subset de queries[3] (filtre solo los de 2 y 3 términos que estén en el vocabulario de su colección) y mida el tiempo de ejecución en cada caso. Para ello, utilice los siguientes patrones booleanos:

   a) Queries |q| = 2
      - t1 AND t2
      - t1 OR t2

[3] https://www.labredes.unlu.edu.ar/sites/www.labredes.unlu.edu.ar/files/site/data/ri/EFF-10K-queries.txt

- t1 NOT t2

b) Queries |q| = 3
   - t1 AND t2 AND t3
   - (t1 OR t2) NOT t3
   - (t1 AND t2) OR t3

¿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?

a)

WIKI SMALL	
Query	Time (sec)
british AND airways	0.0001
british OR airways	0.001
british NOT airways	0.001

b)

WIKI SMALL	
Query	Time (sec)
(hardware AND software) AND english	0.0015
(hardware OR software) NOT english	0.001
(hardware AND software) OR english	0.001

A medida que van aumentando el tamaño de las listas, aumenta también el tiempo en que tarda en procesar la query

4) Codifique un script que empleando la estrategia DAAT sobre el índice creado en el ejercicio 1 resuelva consultas usando el modelo vectorial y retorne los top-k documentos de score mayor.

Se utilizo la estrategia DAAT donde se calcula el score de cada documento por TF*IDF. A continuación, se ven unas querys de ejemplo.

```
Introduce tu consulta: hardware
Top-10 documentos relevantes:
Documento ID: 5619 - Score: 178.87676456704077
Documento ID: 2342 - Score: 119.25117637802718
Documento ID: 1299 - Score: 59.62558818901359
Documento ID: 2537 - Score: 59.62558818901359
Documento ID: 4503 - Score: 59.62558818901359
Documento ID: 524 - Score: 39.75039212600906
Documento ID: 1770 - Score: 39.75039212600906
Documento ID: 2998 - Score: 39.75039212600906
Documento ID: 3754 - Score: 39.75039212600906
Documento ID: 3879 - Score: 39.75039212600906
```

```
Introduce tu consulta: hardware software
Top-10 documentos relevantes:
Documento ID: 5619 - Score: 289.2937253948619
Documento ID: 4648 - Score: 257.63957526491606
Documento ID: 5084 - Score: 245.37102406182478
Documento ID: 2703 - Score: 220.8339216556423
Documento ID: 539 - Score: 179.36636170319065
Documento ID: 2342 - Score: 131.51972758111842
Documento ID: 524 - Score: 101.09314814146526
Documento ID: 1648 - Score: 93.48650328155199
Documento ID: 6014 - Score: 85.87985842163869
Documento ID: 2560 - Score: 81.21795207846073
```

5) 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 3. 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.

   En este punto se implementó un menú para manejar la creación de las skip list, querys y también para recuperar las skip list de un termino dado.

Query	Time (Ej 3)	Time (Skip List)
technical AND school	0.003	0.001
web AND english	0.02	0.001
computer AND week	0.02	0.00027

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

   En la siguiente tabla se puede ver la mejora en la eficiencia en la búsqueda por parte de TAAT ya que estamos usando una colección pequeña.

Query	Query Length	Time using TAAT	Time using DAAT
a0	1	0,001	0,02
a0 musikladen	2	0,001	0,02
deenys nappes tamboril	3	0,001	0,02

7) Comprima el índice del ejercicio 1 utilizando Variable-Length Codes (VByte) para los docIDs y Elias-gamma para las frecuencias (almacene docIDs y frecuencias en archivos separados). Calcule tiempos de compresión/descompresión del índice completo y tamaño resultante en cada caso. Realice dos experimentos, con y sin DGaps. Compare los tamaños de los índices resultantes.

COMPRESIÓN (sin DGaps)	
Time (sec)	
22.25	
Size	
docIDs	17,078 MB
frecuencias	12,899 MB

DESCOMPRESIÓN (sin DGaps)
Time (sec)
6,18

COMPRESIÓN (con DGaps)	
Time (sec)	
22.58	
Size	
docIDs	11,823 MB
frecuencias	12,899 MB

DESCOMPRESIÓN (con DGaps)
Time (sec)
5,57

[4] http://www.labredes.unlu.edu.ar/sites/www.labredes.unlu.edu.ar/files/site/data/ri/queriesDump10K.txt.tar.gz

Podemos ver una reducción importante en el tamaño del archivo de docIDs usando Dgaps (17,078 MB contra 11,823 Mb).

Bibliografía

[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 Sch¨utze. Introduction to Information Retrieval. Cambridge University Press, New York, NY, USA, 2008.