Assignment 8

Information Retrieval and Web Search

Ahmed Rekik - 790063 & Tim Henning - 789242

January 31, 2018

1. Final Code Submission

1.a. Please submit all your code and upload a zip file including also any additional files needed, such as stop word lists. However, you are free to choose whether to use stop word removal, stemming, lemmatization, or neither of these approaches.

`searchengine.py` and `crawler.py` are in the archive. No additional files required (stopwords are downloaded by NLTK).

1.b. Write down the size of your index of the guardian's comments.csv file and a rough approximation for the time you needed to create it.

The size of the index is 19.3 GB and it took 9 hours (AMD A10 Quad Core, 7h indexing and 2h merging). No more than 4-5GB RAM during indexing, 2GB during merging and approx. 1.5GB during query execution should be required.

1.c. Write down the list of files that you create (invertedIndexSeekList.txt, CommentSeekLists.txt, skippingPointer.txt, replyToIndex.txt, authorIndex.txt, etc.) and report their sizes.

The index files are in the `index` directory.

File Name	Size (MB)
postings.dat	18612.0
seek.dat	21.6
reply_seek.dat	91.8
reply_to.dat	243.8
lengths.dat	393.2
stats.dat	103 Byte

2. Query Execution Times

2.a. Report the execution time of your search engine in your local machine and the number of search hits on the guardian's comments.csv file for the following queries. If your search engine does not support some of the optional combinations, simply do not process the query.

Measured Time The time to create the list of the internal IDs of the results was measured, not including the materialization of the real comments (which would just be dominated by disk seeks). Materializing the top 5 comments usually takes constant additional 200-1000ms. The time for phrase queries includes the time to materialize the top 100 results.

Phrase Query Implementation We process phrase queries at the time the comments are materialized. While this works very well to return the top N results for phrase queries in a reasonable amount of time, our search engine can not count how many matching results there are in total (because the materialization of all possible comments is very expensive). For better comparison, if no top N is provided, the top 100 results are calculated and returned.

Query	Search Hits	Execution Time (ms)
election	1309678	12137.12
military conflict	660096	6715.89
'German chancellor'	$100 \leq N \leq 506995$	8115.41
'guardians of the galaxy'	$100 \leq N \leq 1466071$	17113.38
brexit AND economy	23141	5350.68
jared NOT kushner	3527	39.66
isra*	503366	17042.00
ReplyTo:107701851	3	0.06
288 days	2784694	24862.62
merkel NOT chancel*	65505	641.24
eu OR "european union"	Not supported	-
trump AND putin AND merkel AND xi	15	4233.19
'new ye'*	Not supported	-
ReplyTo:107701851 AND 'silicon valley'	Not supported	-

3. Notes on the Implementation

With `-printIdsOnly` the internal IDs are printed. To print the original IDs from the csv file use `-printOriginalIdsOnly` (this takes much longer because the comments have to be read to get the original IDs).

Code Style If we could start again from scratch, we would definitely use a much more modularized code and probably switch to C++. So please excuse the bad code style, this is not representative and we now it is really worse :-/