Your Name: Chih-Wei Chang

Your Andrew ID: cchang3

Homework 1

Collaboration and Originality

Your report must include answers to the following questions:

1. Did you receive help of any kind from anyone in developing your software for this assignment (Yes or No)? It is not necessary to describe discussions with the instructor or TAs.

   I received help from Cheng-Ta Chung. He explained the structure of QryEval package to me, including how each Class is inherited from their parent, such as QryIop and QrySop. He also explained the key difference between a score list and an inverted list to me. Besides, he explained some built-in QryEval functions to me, such as advanceIteratorPast.

2. Did you give help of any kind to anyone in developing their software for this assignment (Yes or No)?

   I discussed several edge cases and pitfalls that might occur when implementing the greedy NEAR algorithm with Teng Zhong. This gave him the basic idea to debug his code.

3. Are you the author of every line of source code submitted for this assignment (Yes or No)? It is not necessary to mention software provided by the instructor.

   Yes.

4. Are you the author of every word of your report (Yes or No)?

   Yes.

Your Name: Chih-Wei Chang

Your Andrew ID: cchang3

Homework 1

1 Structured query set

1.1 Summary of query structuring strategies

Some keywords might have higher chance to occur in certain fields, such as title. In such case, we restrict the keyword to that field. Also, some keywords might have strong correlations, and thus applying a NEAR operator on them might be helpful. On the other hand, we might also try to use an AND operator on such high correlation keywords since this loose the constraint that a NEAR operator might put on our query.

1.2 Structured queries

List your structured queries. For each query, provide a brief (1-2 sentences) discussion of:

1. 69: #OR(sewing instructions.title)
   Instruction is something that will appear in the title to indicate that certain document is an "instruction", thus I put the title constraint on it.
2. 79: #OR(voyager)
   Since there is only one query, and I found that trying different field constraints would not give a better result, I left it unchanged.
3. 84: #NEAR/12(continental plates)
   Continental and plates are two strongly correlated terms, thus I put a NEAR operator on them. I tried different distance and it seems 12 performed reasonably well than others (< 20).
4. 89: #OR(ocd)
   Same reason as the second query. I left it unchanged.
5. 108: #NEAR/2(ralph owen brewster)
   Since this query is very likely meant to search for the person Ralph Owen Brewster, I put a NEAR operator on it to restrict the occurrences of these three terms.
6. 141: #OR(va.keywords #AND(dmv registration))
   Dmv and registration have high correlation, so I put an AND operator on them. Besides, VA might refer to the initial of Virginia, and it would probably be found in a keywords field, so I also put a field constraint on it.
7. 146: #OR(sherwood.keywords #AND(regional library))
   Similarly to sixth query, the user might want to find the regional library, so I put an AND on it. Additionally, Sherwood is something that might appear in the keywords field, so I put an extra field constraint on it.
8. 153: #OR(pocono)
   Same reason as the 2nd and 4th query. I left it unchanged.

9. 171: #NEAR/2(ron howard)
   Same reason as 5th, this might refer to the person name Ron Howard, so I put a NEAR operator on them.
10. 197: #OR(#NEAR/2(idaho state) flower.keywords)
    Idaho State might refer to some kind of flower, so I put these two terms together. Also, the flower is more like a keyword, so I put it in the keywords field.

2 Experimental results

Present the complete set of experimental results. Include the precision and running time results described above. Present these in a tabular form (see below) so that it is easy to compare the results for each algorithm.

2.1 Unranked Boolean

	BOW #OR	**BOW #AND**	**Structured**
P@10	0.0000	0.1100	0.1700
P@20	0.0150	0.1350	0.1900
P@30	0.0200	0.1533	0.1900
MAP	0.0020	0.0665	0.0964
Running Time	8.31 s	1.96 s	3.05 s

2.2 Ranked Boolean

	BOW #OR	**BOW #AND**	**Structured**
P@10	0.1700	0.3700	0.3600
P@20	0.2800	0.4450	0.4750
P@30	0.3367	0.4633	0.4800
MAP	0.1071	0.1882	0.2268
Running Time	9.12 s	2.41 s	4.27 s

3 Analysis of results: Queries and ranking algorithms

Discuss your observations about the differences between the three different approaches to forming queries, and the two different approaches to retrieving documents (i.e., retrieval models) in terms of their retrieval performance and total running time.

Hint: Do not just summarize the results from the previous sections; we can see those results above. You are expected to provide your interpretation of the results based on what you learned in the lectures and readings. This is your chance to show what you learned from this homework assignment - take this section very seriously

Hint: Probably this section doesn't need to be longer than ¾ of a page (not counting these instructions).

Basically, OR operator takes the longer time, while AND takes less. This might result from the difference in the two following stages: matching document ID and sorting results. In particular, AND only match the same document ID, which results in a smaller matching sets. This results in a smaller input for the sorting function, and thus a shorter running time. At the first glance, the results of Unranked Boolean might be contradicted to this observation. However, even Unranked Boolean need to sort the document id and thus it still suffers from the huge amount of input in OR operator. Besides, ranked approach also takes generally longer time than unranked one. The reason is that ranked approach need to calculate the scores, such as term frequency. That being said, from the difference between ranked and unranked one is much smaller than the difference between OR and AND. This might reveal the insight that the sorting could dominate the whole running time rather than calculating the statistics or scores. As for the structured query, it utilize various operators, so naturally its running time is in the middle of pure OR and pure AND.

As for the retrieval performance, pure OR results in poor precision, while pure AND can be quite good at precision. One interesting observation is that, although the MAP of unranked's OR is 1/5 of ranked one, the MAP of unranked's AND doesn't decrease so much. I guess this might be related to the different retrieval size of two operators. AND tends to retrieve smaller amount of document, and thus the "truncated-at-100" might have less effects than OR would have had. On the other hand, structured query performs generally better than the other two. This shows that, when carefully construct the queries, we can get reasonably good results even with such a naïve approach. Also note that, although pure AND outperform structured query at the P10, structured has a consistent better performance.

To sum up, different operators can have huge impact on the running time of search engine. However, simply putting some constraints on queries or combining different operators can significantly ameliorate this issue. Structured queries have not only the advantage on running time, they also perform consistently well than pure queries.

4 Analysis of results: Query operators and fields

Discuss the effectiveness, strengths, and weaknesses of the query operators and fields, and your success and failure at using them in queries. Did they satisfy your expectations?

Hint: Same hints as above.

Different operators yield different retrieval performance, and they also affect the running time in various way. OR tends to retrieve all possible matched documents even for less likely ones, while AND tends to be very strict. These trends become more obvious as the number of queries increase. One possible way to integrate these two operators is first applying AND on some terms which we believe should occur together, and then we apply another OR on this result with other terms. The benefits of doing so is that we can have higher precision on the terms we are confident with and a not-so-bad recall on those terms we are not very sure about. As for NEAR, we can treat it as an augmentation on the AND operator, and thus we can apply same strategy to combine the NEAR and the OR.

Other than the operators, the fields constraint itself can also have great impact on both the retrieval performance and the running time. Take the query #146 as an instance. There is a query term "Sherwood" which is something that might appear in the keywords list instead of the web content body. And thus trying to match it in the body field might not yield a good result, which has a MAP equals to 0 for the pure OR queries. However, if we tried to match it in the keywords field, and also adjust a little bit on the companion terms, we could get a much better retrieval result. Also, as mentioned, the input size for the sorting might dominate the whole running time. Since matching the term in different fields can greatly affect the results size, it can thus affect the running time hugely. However, one thing to bear in mind when using field match is that we should be aware of the possibility that we mis-select the field. For example, some term might only occur in a certain field, e.g. title, and if we tried to match it in the inlink field, we will have poor result. Therefore, although field matching can help us reduce the possible results size, we should still try different combinations first to be sure we are not misusing fields.