Efficient Phrase Querying with an Auxiliary Index
Dirk Bahle, Hugh E. Williams, Justin Zobel

605.744: Information Retrieval
Sabbir Ahmed

Goals and Motivations

Phrase querying refers to querying search engines or other information retrieval systems with a sequence of more than one word. The most frequent number of words in a phrase query is two, and a significant number of phrase queries with more than two words contain common words (or stopwords). Since a significant portion of user queries include phrases, the ever-growing search engines need to evaluate such queries extremely fast.

Inverted indexes are typically helpful in querying single words. The use of nextword indexes has been shown to improve support for querying phrases, albeit at the expense of larger storage requirements. This paper proposes a method to optimize querying for phrases by using a combination of inverted indexes and nextword indexes.

Background

Inverted indexes are the standard method method for supporting querying on large corpuses. Typically, they are represented as B-trees of indexes mapped to their postings of triples that contain the document identifier, the term frequency in that document, and the positions of the term in those documents.

Nextword indexes consist of data structures that are relatively more expansive, where each term, or firstword, maps to data structures of its corresponding nextword. Each of the nextwords then maps to their posting lists of where the firstword-nextword pairs occur. The sizes of such structures increase much faster than conventional inverted indexes, easily surpassing twice the storage requirements.

With the added complexity of nextword indexes, it may appear that they may not be necessary to accomplish what inverted indexes can achieve using less space. Phrase querying in inverted indexes does involve additional steps of sorting the lists from the rarest to the most common terms and then merging those lists. However, inverted index files reward stopword removal for smaller storage requirements and faster retrievals. Stopword removals contribute to a loss of accuracy for phrase querying, where phrases such as “end of days” or “the Who” get removed entirely.

Nextword indexes perform much better with stopwords present in phrases. Their postings lists are typically short since most firstword-nextword pairs only occur infrequently. Using the example from before, the pair “the”-“who” is significantly less frequent than those terms occurring in an inverted index.

Naturally, it appears that there needs to be an implementation where both the space efficiency of inverted indexes and the accuracy and speed of nextword indexes with stopwords can be achieved,

Approach

The authors suggest a combination of inverted indexes with an auxiliary nextword for terms that are considered stopwords. They also propose a constraint on the nextword terms, where only

stopwords can be used as firstwords in the pairs. This approach forces the engine to switch its querying method based on individual terms in the phrase; if a term is infrequent enough to not be considered a stopword, then it will be searched in the inverted index for its postings list. On the other hand, if the system encounters a stopword in the phrase, then it will be considered the firstword and its nextword will be searched. Once a match is found for the stopword and the subsequent term, its postings list will be retrieved. The authors have laid out the algorithm in the following steps:

1. Identify each terms of the phrase to determine if they should be searched for their nextword, or if they exist in the inverted index in the standard approach
2. If a term is not considered a stopword, then retrieve its postings list
3. If a term is considered a stopword, then lookahead to the next term and search for the firstword-nextword pair in the nextword index
4. For all terms not in a firstword-nextword pair, sorting
5. Process the postings lists in increasing order of firstword frequency

Experiments and Results

The authors used the publicly available query logs provided by Excite to calculate the statistics used to benchmark in this paper. The logs included almost 1.6 million queries from 1997 to 1999, where the median number of words in the phrase queries were 2 and the mean was almost 2.5. The authors' test dataset was extracted from the TREC large web track, containing about 8.3 GB of text.

In the final results where the evaluation times were calculated, the number of stopwords used in the nextword index proved to be significant. Including all of the 254 recognized stopwords in the dataset, the combined index showed a 70% improvement in the overall query evaluation. This improvement is outperformed when only using a standard inverted index which yields an improvement of 88.5%, but with the cost of accuracy. By adding the auxiliary nextword index that is 3% of the size of its corresponding inverted index, similar results can be achieved without compromising accuracy.

Discussion

One major issue that can be brought up with the paper is its relevancy to the properties of modern queries. The logs used to generate the benchmarks are from 1999, and web searching behaviors have changed drastically over the last 2 decades. It can be argued that users have become more literate with phrase queries and the average number of words in a query is over 5. Does appending an auxiliary nextword index to a standard inverted index scale to modern queries?

Another issue I personally had with the paper's findings was that there were no metrics to compare accuracies before and after adding the auxiliary nextword index. Sacrificing a relatively smaller storage overhead for accuracy was used as one of the main benefits of their proposal, but the authors did not provide any evidence to support the comparison.