2009 International Conference on Advanced Information Networking and Applications Workshops

Document-Oriented Pruning of the Inverted Index in Information Retrieval Systems

Lei Zheng and Ingemar J. Cox
University College London
Gower Street, London, WC1E 6BT, United Kingdom
lei.zheng@ucl.ac.uk, ingemar@ieee.org

Abstract

Searching very large collections can be costly in both computation and storage. To reduce this cost, recent research has focused on reducing the size (pruning) of the inverted index. The inverted index represents a table, the rows and columns of which are terms in the lexicon and documents in the collection, respectively. A non-zero entry in the table, known as a posting, indicates that the corresponding document contains the term. Previous researches on static index pruning was either (i) posting-oriented, in which less important postings are removed from the table, or (ii) term-oriented, in which less important terms are removed from the table. In this paper, we investigate a new, document-oriented pruning strategy that removes entire columns of the table, i.e. removes less important documents from the collection. Three methods for estimating the importance of a document are proposed. Methods 1 and 2 are dependent on the score function of the retrieval system (e.g. Okapi BM25), while Method 3 is independent of the retrieval system. Experimental results compare the three proposed methods with Carmel et al.'s posting-oriented approach, using both the FT and LA Times collections and using both ordinary and difficult queries. Based on mean average precision and precision at 10, experimental results show that Method 3 generally performs best on the FT collection for pruned indexes down to 35% of the original size. However, for more severe pruning, Carmel et al.'s algorithm is better. For the LA Times collection, the performance of Method 3 and that of Carmel et al. are reversed. This variation in performance across collections has not been previously reported.

1. Introduction

Web search has been critical to the success of the Web. Search engines are often the first point of entry for users. It is well known that search engines can generate significant revenue from paid placement advertising. However, less well known is the fact that Web search can be very costly to provide. It is estimated that Google and Microsoft currently spend more than 1 billion U.S. dollars per annum to provide

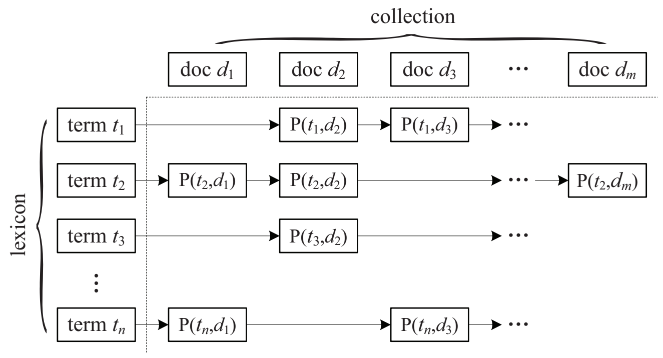

Figure 1. An example of an inverted index for information retrieval systems

the computational, storage and network resources necessary to support their services. Consequently, there is considerable interest in reducing these costs, as even relatively small improvements can produce large savings.

Web search involves the indexing of an enormous number of documents. Therefore, one focus of attention has been on reducing the size of the inverted index. An inverted index is a data structure that is commonly used to implement information retrieval (IR) [1]. Figure 1 gives an example of the structure of an inverted index. Conceptually, it represents a table, the rows and columns of which are terms (words) in the lexicon (vocabulary) and documents in the collection, respectively. An entry in the table, known as a posting, indicates that the corresponding document contains the term. In practice, the inverted index only records non-zero entries. Each row of the index consists of a posting list, where each posting is a pointer to a document containing the term.

Index pruning can be either dynamic or static. Dynamic pruning [2], [3] decides during query processing, whether certain terms or postings are worth adding to the accumulated retrieval scores, and whether the ranking process should continue or stop. By contrast, static pruning [3] removes entries from the index in advance (of any query), therefore reducing the index size. The focus of this paper is on static index pruning, which can be considered equivalent to lossy compression of the inverted index. Since information is discarded (lost) as part of the pruning (compression) process, it is necessary to examine how the performance of the IR system is affected by the pruning strategy. Besides lossy compression, lossless compression is also possible. Ntoulas and Cho [4] described a structure for a two-tiered cache of the inverted index, in which the first tier stores the pruned index and the second tier stores a full index.

Carmel *et al.* [3] first introduced the concept of static index pruning and described a *posting-oriented pruning* strategy. Their approach, described in Section 2, removes less important postings from the inverted index, thereby reducing its size. The basis of this strategy is to sparsify the index table, i.e. to decide which of the individual postings should remain in the index.

Blanco and Barreiro [5] extended the work of static index pruning, by considering a *term-oriented pruning* strategy. The feature of such strategy is to eliminate rows from the index table, i.e. to decide which of the terms in the lexicon should remain in the index. Their work was inspired by the concept of stopwords. Blanco and Barreiro studied how performance is affected by the removal of less important terms from the lexicon. Their experiments demonstrated that this approach can perform competitively to Carmel's posting-oriented pruning.

In this paper, we present a new perspective to prune the inverted index, i.e. *document-oriented pruning*. Our strategy is to decide whether all postings pointing to a specific document should remain in the index. This is equivalent to eliminating entire columns from the index table. Our work is based on the assumption that not every document is equally important in a collection (just as not every term is equally important in the lexcion). Recent research on the "findabilty" of documents [6], [7] also suggests that some documents in a collection are unlikely to be retrieved by the IR system. Three algorithms for statically scoring the importance of documents are described, two of which are dependent on the score function of the retrieval system, while the third is independent of the retrieval system.

The remainder of this paper is organized as follows: Section 2 describes our baseline algorithm, i.e. Carmel's posting-oriented pruning technique. Section 3 then presents our document-oriented pruning algorithms. The experiments and results are given in Section 4. Finally, Section 5 provides a discussion of our results and directions for future work.

2. Carmel's Posting-Oriented Index Pruning

Carmel *et al.* [3] introduced the concept of static index pruning, and described a posting-oriented pruning strategy that removes less important postings from an inverted index.

The importance score of a posting $\mathrm{P}(t, d)$[1] is determined by the score function of the retrieval system. For the term frequency inverse document frequency (TFIDF) of the SMART retrieval system [8], the importance score of a posting is given by

$$\mathrm{P}(t, d) = \frac{\frac{\log(1+tf(t))}{\log(1+avgtf)} \log\left(\frac{N}{N_t}\right)}{|d|} \quad (1)$$

where $tf(t)$ is term t's frequency in document d, $avgtf$ is the average term frequency for document d, N is the number of documents in the collection, N_t is the number of documents containing term t, and $|d|$ is the length of document d.

Carmel's pruning removes less important postings, by setting some non-zero table entries to zero. They described two closely related methods, "uniform pruning" and "term-based pruning". For uniform pruning, the pruning threshold τ is the same for all terms, while for term-based pruning, the threshold τ_t is selected for each term t. The posting of the pruned index $\mathrm{P}^*(t, d)$ is equal to $\mathrm{P}(t, d)$ if the score is greater than the threshold and zero otherwise.

For term-based pruning, Carmel *et al.* described two algorithms, i.e. "top-k pruning" and "δ-top pruning", to define the pruning threshold τ_t for each term t. The threshold of top-k pruning is chosen by $\tau_t = \epsilon \cdot z_t$, where z_t is the term's kth highest posting score, and ϵ is a parameter used to control the pruning rate. Instead of the kth highest posting score, δ-top pruning uses another parameter δ times the highest score z'_t to define the pruning threshold, i.e. $\tau_t = \epsilon \cdot \delta z'_t$. The nature of top-$k$ pruning and δ-top pruning are the same. The only difference is that whether the pruning threshold is determined by the kth highest posting score z_t or δ times the highest score z'_t. In this paper, we implement Carmel's top-k pruning as our baseline algorithm.

3. Document-Oriented Index Pruning

In this Section, we discuss our document-oriented pruning. Our strategy is to decide whether all postings pointing to a specific document should remain in the index, which is equivalent to eliminating entire columns from the index table. Our decision on which documents to remove is made according to their importance scores. Three methods for statically scoring documents are described, two of which are dependent on the score function of the retrieval system, while the third is independent of the retrieval system.

3.1. Method 1

First of all, we compute the importance score for all the postings $\mathrm{P}(t_i, d_j)$ in the inverted index, by using the

1. In [3], Carmel *et al.* used the symbol $\mathrm{A}(t, d)$.

score function of the retrieval system. Posting score $\mathrm{P}(t_i, d_j)$ reflects term t_i's contribution to document d_j. In the sense of information retrieval, a posting with a high score is more discriminative (important) than the one with a low score. Thus, the value of $\mathrm{P}(t_i, d_j)$ reflects the importance of the posting.

After assigning every posting an importance score according to the retrieval system, we score documents by averaging all the associated posting scores. Thus, Method 1 for scoring document d is

$$S_1(d) = \frac{1}{|d|} \sum_{t_i \in d} tf(t_i) \cdot \mathrm{P}(t_i, d) \tag{2}$$

The purpose of the denominator, $|d|$, is to reduce the affect of different document lengths.

We assume that an important document should normally contain more discriminative postings than a less important document. In other words, the higher the document score, the more important of the document is. Therefore, our pruning strategy removes low scoring documents, i.e. all the postings associated with such documents. Note that the score of a document is an average over all its postings. Therefore, a document can be removed, even if some individual postings have high scores, but the overall document score is below the threshold.

3.2. Method 2

In Method 1, the posting scores of the same term are different across documents. In Method 2, we first average the individual posting scores for a term t_i, to obtain an estimate of t_i's overall importance $\mathrm{T}(t_i)$. In other words, $\mathrm{T}(t_i)$ is a constant across all documents containing t_i, i.e.

$$\mathrm{T}(t_i) = \frac{1}{N_{t_i}} \sum_{d_j} \mathrm{P}(t_i, d_j) \tag{3}$$

where N_{t_i} is the number of documents containing term t_i.

The document score of Method 2, is then an average of the associated term scores, i.e.

$$\begin{aligned} S_2(d) &= \frac{1}{|d|} \sum_{t_i \in d} tf(t_i) \cdot \mathrm{T}(t_i) \\ &= \frac{1}{|d|} \sum_{t_i \in d} \left(\frac{tf(t_i)}{N_{t_i}} \sum_{d_j} \mathrm{P}(t_i, d_j) \right) \end{aligned} \tag{4}$$

3.3. Method 3

In the previous two methods, the importance score is a function of the specific retrieval system. Thus, for example, if we change our retrieval system from the Okapi BM25 probabilistic model to the language modeling approach, a document's score will be changed as well. However, a score that is independent of the retrieval system may more reliably reflect the importance of a document.

Inverse document frequency (IDF) is widely used as a measure of a term's importance. It is defined [9] as the logarithmic ratio of the total number of documents in a collection, N, to the number of documents containing the term, N_{t_i}, i.e.

$$\mathrm{IDF}(t_i) = \log\left(\frac{N}{N_{t_i}}\right) \tag{5}$$

Common words, such as "the", "and", "it", are likely to appear in every document among the collection. Thus, they have low IDF values. Conversely, terms that only occur in some small number of documents have correspondingly high IDF values.

The most common form of IDF weighting was introduced by Robertson and Sparck-Jones [10]. It normalizes with respect to the number of documents not containing the term $(N - N_{t_i})$ and adds a constant 0.5 to both the numerator and the denominator in order to moderate extreme values. The normalized inverse document frequency (NIDF) is defined by

$$\mathrm{NIDF}(t_i) = \log\left(\frac{N - N_{t_i} + 0.5}{N_{t_i} + 0.5}\right) \tag{6}$$

We assume that documents containing frequently occurring terms are less important than documents containing less frequently occurring terms. Thus, Method 3 for scoring documents replaces $\mathrm{T}(t_i)$ in Equation 4 with $\mathrm{NIDF}(t_i)$, i.e.

$$\begin{aligned} S_3(d) &= \frac{1}{|d|} \sum_{t_i \in d} tf(t_i) \cdot \mathrm{NIDF}(t_i) \\ &= \frac{1}{|d|} \sum_{t_i \in d} tf(t_i) \cdot \log\left(\frac{N - N_{t_i} + 0.5}{N_{t_i} + 0.5}\right) \end{aligned} \tag{7}$$

4. Experimental Evaluation

In this Section, we first describe our experimental setup, followed by experiments that compare the three proposed methods with the baseline algorithm of Carmel *et al.*

4.1. Experimental setup

All our experiments are carried out on the LEMUR toolkit [11]. Documents are stemmed using the Krovetz stemmer [12], but stopwords are not removed in the initial stage. This is because we need a full index to compute our importance score for each document. Instead, stopwords are removed after pruning. We use the stopword list suggested by Fox [13], which includes a total of 421 stopwords.

In our evaluations, the "title" part and the "description" part of TREC topics are used as evaluation queries. In all our experiments, Okapi BM25 [14] is used as the score

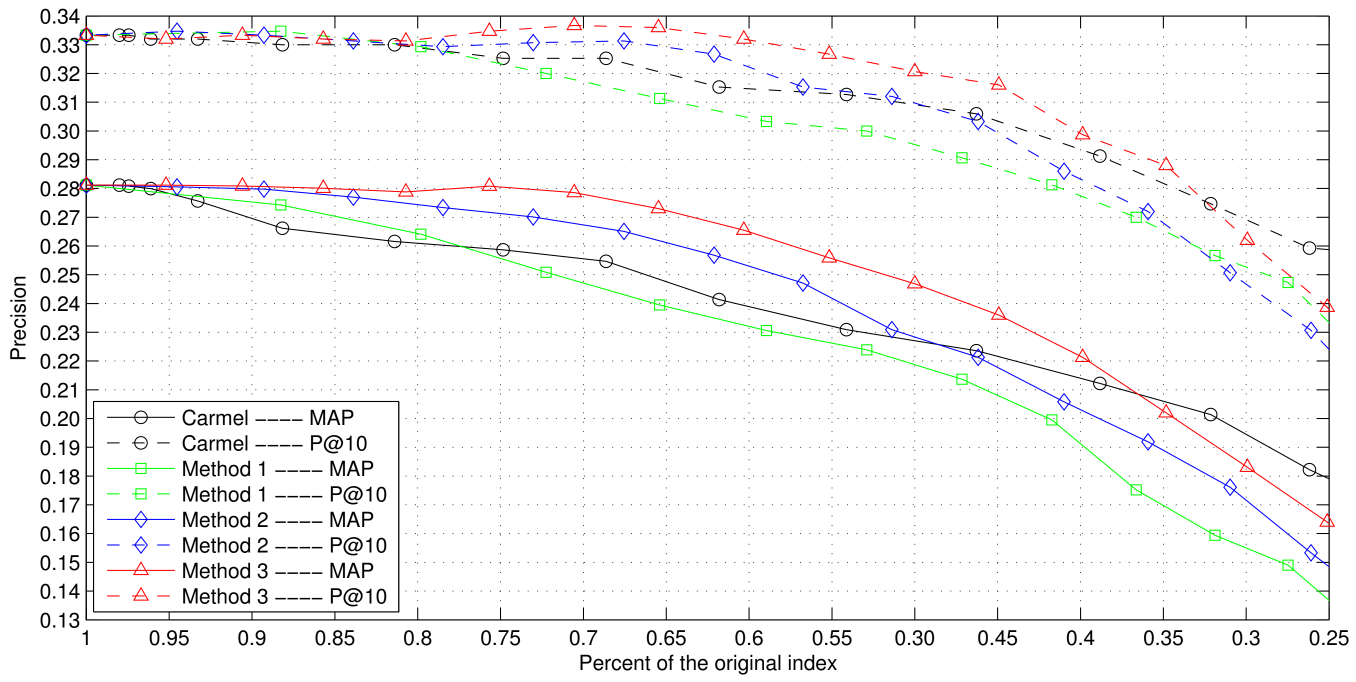

Figure 2. Performance as a function of various levels of pruning. Results are for the Financial Times collection (evaluated by topics 301-450)

function between a query q and a document d. There are three parameters in the BM25 scoring function, i.e. b, k_1, and k_3. We use the recommended values [14] as $b = 0.75$, $k_1 = 1.2$, and $k_3 = 1000$, respectively.

Carmel's top-k pruning algorithm is implemented as our baseline. The value of k is set to 10 as in [3], and the different pruning levels are obtained by modifying the parameter ϵ. We use the same measures as in [3], i.e. mean average precision (MAP) and precision at 10 (P@10), to evaluate the retrieval performance for different pruning levels. As in [3], the percent of the index is defined as the ratio of the number of postings in the pruned index to that in the original index.

4.2. Experiment 1: FT with topics 301-450

We first compare our document-oriented pruning with Carmel's pruning on the Financial Times (FT) collection consisting of 210,158 documents. TREC 6, 7 and 8 ad hoc topics (i.e. topics 301-450) are used to evaluate the performance.

Figure 2 shows the experimental result. We observe that for P@10 (top set of curves), the performance of all four algorithms is similar and is almost unaffected by low levels of pruning (i.e. for the size of pruned indexes greater than 80% of the original index).

As the pruning increases, Carmel's algorithm and Method 1, show slight performance degradations, while Methods 2 and 3 actually exhibit small performance improvements. That is, the performance of Methods 2 and 3 is actually better than the original unpruned index. Although perhaps unexpected, clearly if, for the unpruned index, irrelevant documents are retrieved in the top-10, and these documents are subsequently pruned, then P@10 can actually improve.

For pruned indexes of less than 60% of the original size, performance of all four algorithms begins to degrade below that measured for no pruning. However, the degradation is relatively smooth. It is observed that Method 3 exhibits superior performance across all four methods, until pruning reduces the index to less than 35% of its original size, after which Carmel's method performs best.

For MAP measure (bottom set of curves), the trend of curves behaves similar to that of P@10 measure.

4.3. Experiment 2: FT with difficult topics

In the TREC 2003 and 2004 robust tasks, NIST selected 50 difficult topics to evaluate the robustness (reliability) of an IR system. In our experiment, difficult queries help us to understand whether our pruning techniques are stable for both ordinary queries (that are tested in Experiment 1) and difficult queries. In this experiment, we use the same document collection as in Experiment 1, but difficult topics as evaluation queries.

The results are provided in Figure 3. As expected, the

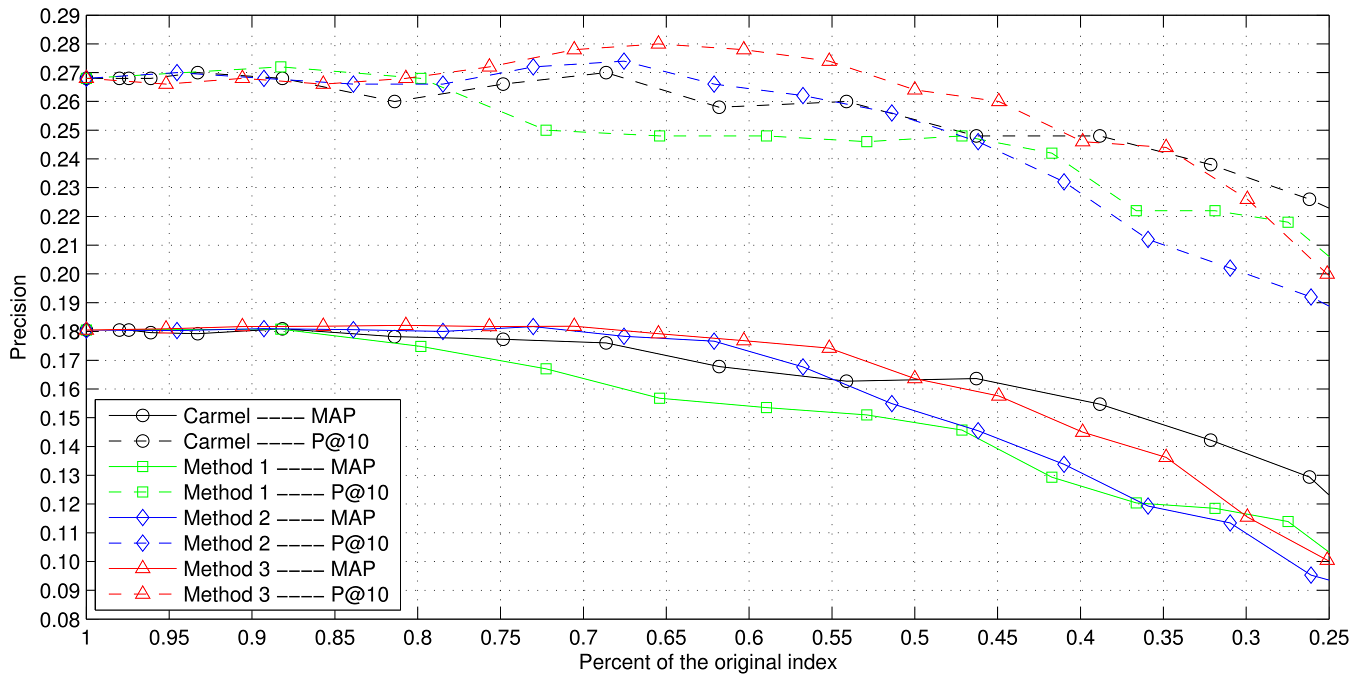

Figure 3. Performance as a function of various levels of pruning. Results are for the Financial Times collection (evaluated by 50 difficult topics)

absolute values of P@10 and MAP decline significantly for all four algorithms, since these are, after all, difficult queries. However, the relative performance for all four algorithms generally remains very similar.

4.4. Experiment 3: LA with topics 401-450

Besides different types of queries, the performance across different collections is also examined. We now use the same document collection and the same evaluation queries as used in [3]. The document collection is the Los Angeles (LA) Times consisting of 131,896 documents. TREC 8 ad hoc topics (i.e. topics 401-450) are used as evaluation queries.

The results of Figure 4 show that the relative performance of Method 3 and Carmel's method is swapped. Generally, Carmel's algorithm exhibits superior performance, while Method 3 exhibits the best performance of our three proposed algorithms. Such variability in performance across collections has not previously been reported and further investigation is needed to fully understand this.

5. Conclusions and Future Work

Previous work on static index pruning has either pruned postings or pruned terms. In this paper, we investigate a third possibility, that of pruning documents. Although this approach may seem somewhat counterintuitive, we observe that in some scenarios our algorithms are competitive or better than posting-oriented pruning. Three methods for determining which documents to remove are proposed. The first two are dependent on the score function of the retrieval system, while the third is independent of the retrieval system.

Experimental results on the FT collection show that Method 3, which is independent of the retrieval system, exhibits superior performance for indexes that are pruned down to 35% of the original size. However, if further pruning is performed, Carmel's method has better performance. The relative performance of the four algorithms is unaffected when experiment is repeated on difficult queries from the robust track of TREC. However, the relative performance is significantly affected by changing the collection to the LA Times. In this case, the relative performance of Method 3 and Carmel's method is swapped, and Carmel's method is observed to exhibit best performance. Method 3 remains the best of the three algorithms we proposed.

The variability of performance across collections has not previously been reported and is clearly one avenue for future work. In particular, we intend to examine the Web track of TREC (WT10G and .Gov) in order to experiment with much larger collections.

We also note that our methods are complementary to that of Carmel. In fact, when a posting is removed, it does not necessarily remove the document from the collection, as other postings may still point to that document. It may

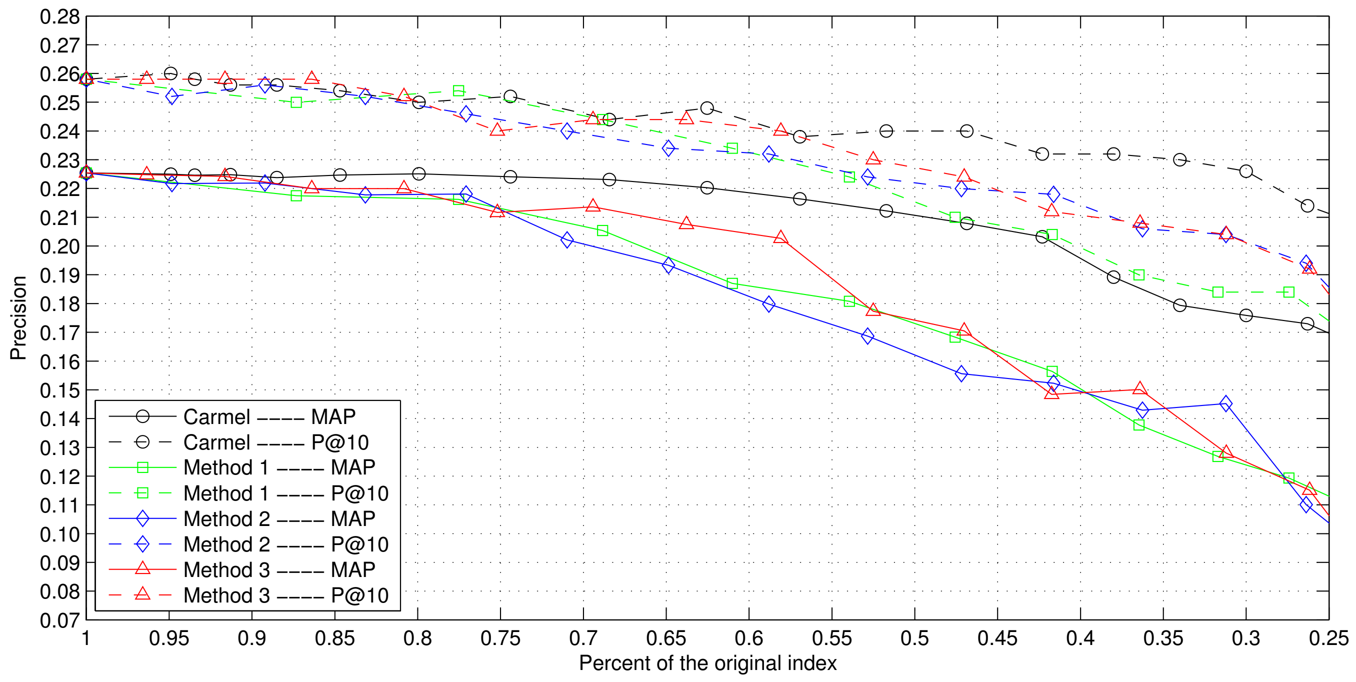

Figure 4. Performance as a function of various levels of pruning. Results are for the Los Angeles Times collection (evaluated by topics 401-450)

therefore be advantageous to consider a combined approach that prunes both postings and documents.

References

[1] C. D. Manning, P. Raghavan, and H. Schtze, *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[2] H. Turtle and J. Flood, “Query evaluation: strategies and optimizations,” *Information Processing and Management*, vol. 31, no. 6, pp. 831–850, 1995.

[3] D. Carmel, D. Cohen, R. Fagin, E. Farchi, M. Herscovici, Y. S. Maarek, and A. Soffer, “Static index pruning for information retrieval systems,” *Proceedings of Special Interest Group on Information Retrieval (SIGIR’01)*, pp. 43–50, September 2001.

[4] A. Ntoulas and J. Cho, “Pruning policies for two-tiered inverted index with correctness guarantee,” *Proceedings of Special Interest Group on Information Retrieval (SIGIR’07)*, pp. 191–198, July 2007.

[5] R. Blanco and A. Barreiro, “Static pruning of terms in inverted files,” *Proceedings of European Conference on Information Retrieval (ECIR’07)*, pp. 64–75, 2007.

[6] L. Azzopardi and V. Vinay, “Accessibility in information retrieval,” *Proceedings of European Conference on Information Retrieval (ECIR’08)*, pp. 482–489, 2008.

[7] ——, “Retrievability: An evaluation measure for higher order information access tasks,” *Proceedings of Conference on Information and Knowledge Management (CIKM’08)*, pp. 561–570, 2008.

[8] G. Salton, *The SMART Retrieval System - Experiments in Automatic Document Processing*. Prentice Hall Incorporated, 1971.

[9] K. Sparck-Jones, “Index term weighting,” *Information Storage and Retrieval*, vol. 9, no. 11, pp. 619–633, 1973.

[10] S. E. Robertson and K. Sparck-Jones, “Relevance weighting of search terms,” *Journal of the American Society for Information Science*, vol. 27, no. 3, pp. 129–146, 1976.

[11] P. Ogilvie and J. Callan, “Experiments using the lemur toolkit,” *Proceedings of the Tenth Text Retrieval Conference (TREC-10)*, 2001.

[12] R. Krovetz, “Viewing morphology as an inference process,” *Proceedings of Special Interest Group on Information Retrieval (SIGIR’93)*, pp. 191–202, 1993.

[13] C. Fox, “A stop list for general text,” *ACM Special Interest Group on Information Retrieval Forum*, vol. 24, no. 1-2, pp. 19–21, 1990.

[14] K. Sparck-Jones, S. Walker, and S. E. Robertson, “A probabilistic model of information retrieval: development and comparative experiments,” *Information Processing and Management*, vol. 36, no. 6, pp. 779–808, 2000.