Fast Automatic Passage Ranking*
(MultiText Experiments for TREC-8)

G. V. Cormack* C. L. A. Clarke* C. R. Palmer+ D. I. E. Kisman*

* Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada
+ School of Computer Science, Carnegie Mellon University, Pittsburgh

1 Introduction

TREC-8 represents the fifth year that the MultiText project has participated in TREC [2, 1, 4, 5].

The MultiText project develops and prototypes scalable technologies for parallel information retrieval systems implemented on networks of workstations. Research issues are addressed in the context of this parallel architecture. Issues of concern to the MultiText Project include data distribution, load balancing, fast update, fault tolerance, document structure, relevance ranking, and user interaction.

The MultiText system incorporates a unique technique for arbitrary passage retrieval. Since our initial participation in TREC-4 our TREC work has explored variants of this technique.

For TREC-8 we focused our efforts on the Web track. In addition, we submitted runs for the Adhoc task (title and title+description) and a run for the Question Answering task.

2 Arbitrary Passage Retrieval

All our experiments are based on similar passage retrieval techniques. Passages identified by this technique are used in different ways in different experiments. Versions of the technique have been described elsewhere [4, 3]. This section provides a brief, up-to-date description of the technique as used in our TREC-8 experiments.

Each document D in a database is treated as an ordered sequence of terms

$$D = d_1\ d_2\ d_3 ... d_m,$$

*Email regarding this article may be sent to mt@plg.uwaterloo.ca. The MultiText project is funded by Communications and Information Technology Ontario.

and a query is treated a set of terms

$$Q = \{q_1,\ q_2,\ q_3, ...\}.$$

An *extent* (u, v), with $1 \le u \le v \le m$ is used to represent the subsequence of D beginning at position u and ending at position v

$$d_u\ d_{u+1}\ d_{u+2} ... d_v.$$

An extent (u, v) *satisfies* a term set $T \subseteq Q$ if the subsequence of D defined by the extent contains all the terms from T. That is,

$$|\,\{t \mid t \in T \text{ and } t \in \{d_u\ d_{u+1}\ ...d_v\}\}\,| \;=\; |T|.$$

An extent (u, v) is a *cover* for T if (u, v) satisfies T and the subsequence corresponding to (u, v) contains no subsequence that also satisfies T. That is, there does not exist an extent (u', v') with either $u < u' \le v' \le v$ or $u \le u' \le v' < v$ that satisfies T.

We generalize the notion of a cover as follows: An extent (u, v) *n-satisfies* a term set $T \subseteq Q$ if the subsequence of D defined by the extent contains exactly n terms from T. That is,

$$|\,\{t \mid t \in T \text{ and } t \in \{d_u\ d_{u+1}\ ...d_v\}\}\,| \;=\; n.$$

An extent (u, v) is an *n-cover* for T if (u, v) n-satisfies T and the subsequence corresponding to (u, v) contains no subsequence that also n-satisfies T.

The MultiText System uses a fast algorithm to compute n-covers over all documents in a collection [3]. Passages are assigned scores based on their lengths and on the weights assigned to the query terms contained within them. A term t is assigned an IDF-like weight

$$w_t \;=\; \log(N/f_t),$$

where f_t is the number of times that t appears in the database and N is the total number of term positions in the database (or equivalently, N is the sum of the lengths of all the documents in the database). A standard IDF weight is not used since in-document frequencies are not stored in the MultiText index.

The weight assigned to a set of terms $T \subseteq Q$ is the sum of the weights assigned to each term in T

$$W(T) \;=\; \sum_{t \in T} w_t.$$

A score is assigned to an extent based on its length

$$I(p, q) \;=\; \begin{cases} \frac{\mathcal{K}}{q-p+1} & \text{if } q - p + 1 \ge \mathcal{K} \\ 1 & \text{if } q - p + 1 \le \mathcal{K} \end{cases} \tag{1}$$

where $\mathcal{K}$ is a *cutoff* parameter set to values between 1 and 16. If an extent (u, v) is a n-cover for the term set T then it can be assigned a score combining the length of the extent and the weight of the terms contained within it

$$C(T, u, v) = W(T') \;+\; |T'| \log(I(u, v)) \tag{2}$$

where $T' \subseteq T$ is the set of terms from T contained in term sequence associated with (u, v).

Method	S-stem?	Fast Adhoc	Small Web	Large Web
1	yes	0.2143 (0.3410)[1]	0.3066 (0.3620)[2]	
2	yes	0.2233 (0.3550)	0.3203 (0.3800)	0.4783 (0.5650)[3]
2	no	0.2126 (0.3390)	0.3029 (0.3640)	0.4869 (0.5720)[4]
2*	no	0.2233 (0.3550)	0.3203 (0.3800)	0.4704 (0.5580)[5]

*max. 3 terms

[1] `uwmt8a0` [2] `uwmt8w0` [3] `uwmt8lw0` [4] `uwmt8lw1` [5] `uwmt8lw2`

Figure 1: Average Precision (Precision @20) for Web Track and Fast Adhoc Runs.

3 Web Track and Fast Adhoc

The main objective of our Web Track participation was to achieve very fast high-precision retrieval, particularly on the 100 GB Large Web Track Corpus. The general approach is to derive automatically a tiered query - a sequence of successively weaker sets of search terms, which are applied in order until a sufficient number of distinct documents are retrieved to satisfy the task requirements (1000 documents for Adhoc and Small Web; 20 documents for Large Web).

For efficient retrieval we chose to use very small sets of search terms - three or fewer in most cases, and never more than five. For the Adhoc and Small Web tasks we used the title words; for the Large Web task we used only words from the query with stopwords eliminated. Efficiency (and, serindipitously, precision) is improved when passages are restricted to a maximum length (128 words) and the set of search terms rarely coincide within a passage.

Thus we avoid typical sources of inefficiency for retrieval methods based on a large number of weighted terms: we use only a small set of terms in the first place, and we consider only those passages in which exactly these terms co-occur. In contrast, traditional implementations based on weighted terms involve computing the score of every document or passage that contains any of the terms.

In TREC 7, we demonstrated that very good precision and efficiency can be achieved with manually selected sets of terms. Poorer precision resulted when the sets of terms were selected automatically. For TREC 8, our objective was to improve the precision of automatic term selection in tiered queries.

The offical runs for the Adhoc and Small Web tasks were due at NIST several weeks before those for the Large Web task. During this interval, we refined our approach to automatic tiering. For this reason, our official fast Adhoc and Small Web runs use the same method (Method 1) while the Large Web run uses a different method (Method 2).

We later applied Method 2 to the Adhoc and Small Web tasks, using NIST's qrels which were published after the official runs were judged. Although these runs are at a disadvantage relative to official runs, Method 2 achieved better precision in all tasks. Average precision and precision @20 for these runs are summarized in Figure 1.

3.1 Tiering Methods

Both tiering methods approximate the ranking formula $C(T, u, v)$ given above as formula 2. A direct computation of $C(T, u, v)$ was rejected for efficiency reasons: such a computation would require that all passages containing any subset of the terms be evaluated. Instead we fix the set of terms in each tier, and compute $I(p, q)$ for each interval of length 128 or less which contains exactly these terms.

Each tier uses as a set of terms T', a subset of T, the initial terms. These subsets are ordered so that those likely to contribute to a high value of the ranking formula $C(T, u, v)$ are considered first. Method 1 and Method 2 differ in the ordering of tiers.

We note that formula 2 approximates the (logarithm of the) probability that the terms of T' coincide within a passage of length $v - u + 1$, assuming that the term occurrences are uniformly distributed in the corpus.

Method 1 uses instead the actual number of occurrences of passages containing the terms in T'. That is, Method 1 is a two-pass method: the query is evaluated for all $T' \subseteq T$. For each T', we compute $N(T')$, the number of passages of length 128 or less that contain all the terms of T'. Tiers are weighted by the formula $\log_2(N/N(T'))$. Tiers are ordered by weight, with tiers having similar weights combined by disjunction. More specifically, the first tier is the one with the highest weight (it will necessarily have $T' = T$) and it is combined with all others whose weight is not more than 1 less. The next tier is chosen and combined with subsequent tiers whose weight is within 1, and so on. Within tiers, documents are ranked by formula 1 with $\mathcal{K} = 4$. Because it requires two passes and examines all passages containing all combinations of the query terms, Method 1 is not suitable for very fast retrieval from large corpora. For this reason, we developed Method 2 for the Large Web task.

Method 2 approximates a priori the number of passages containing the terms of T', assuming the terms are uniformly distributed. This approximation is effected by substituting in formula 2 representative values of u and v such that $u - v + 1 = 128$; that is, the longest acceptable passage containing the terms of T'. As with Method 1, we use $\mathcal{K} = 4$ in formula 1, and combine tiers whose weights differ by less than 1.

For the Large Web task, we used three variants of Method 2: `uwmt8lw0` takes as T the set of query words, unstemmed with stopwords removed; `uwmt8lw1` takes as T only the three lowest-frequency 3 words from the query; `uwmt8lw2` uses the set of query words, with S-stemming applied to those having plural suffixes. For example the query term "goats" would be expanded to {"goat","goats"} but the term "cow" would be left unexpanded. This method of S-stemming was also applied to the Adhoc and Small Web runs.

Method 2, with and without S-stemming, was applied to the Adhoc and Small Web tasks, yielding the results in Figure 1. The methods used in each row are identical. It was not feasible to apply Method 1 to the Large Web task - no complete set of judgements is available with which to evaluate precision. The methods of `uwmt8lw0` and `uwmt8lw2` yield identical results when applied to the other two tasks, as there are at most three terms in the title field, from which we drew T.

We note that Method 2 outperforms Method 1 in all runs. S-stemming, on the other hand, improves performance on Adhoc and Small Web, while degrading performance on Large Web.

3.2 Large Web Performance

The Large Web runs were accomplished on two P2-350 computers; the exact configuration was demonstrated at the SIGIR 99 and TREC 8 conferences. Each system runs four copies of a single-threaded search engine, thus achieving CPU/IO overlap and overlap in access among the four disk drives. Each tier is sent to all engines, and all engines respond with the requested number of documents. These are merged by score to produce the combined result. Our choice of two machines was arbitrary: we wanted to build a portable version that we could demonstrate and the index and data fit easily on two machines. Had we used more computers we could have achieved much faster processing times; we have observed in previous experiments that distributing the data yields a linear improvement in query speed. Eventually, constant communication and query setup costs dominate retrieval time; we estimate these costs to be at least an order of magnitude less than the times reported here.

Execution times for the three Large Web runs are as follows: `uwmt8lw0`: 0.841 sec/query; `uwmt8lw1`: 0.735 sec/query; `uwmt8lw2`: 1.010 sec/query. These times were bettered only by AT&T (0.516 seconds/query; 0.354 P@20) and Fujitsu (0.54 sec/query; 0.507 P@20). The only conclusion that can be drawn is that these times are of the same order, although they are achieved on very different hardware platforms.

`uwmt8lw0` and `uwmt8lw2` achieved the best average precision and precision @20 of all runs. `uwmt8lw1` was edged out by a run from Microsoft/City University (1.62 sec/query; 0.561 P@20).

Hawking et al[6] give full comparative results for the Web Tracks.

4 Other Experiments

4.1 Other Adhoc Experiments

One of our Adhoc runs (`uwmt8a0`) was a repeat of our Small Web Track run using the same ranking method. This run was discussed in Section 3.

For our two other Adhoc runs, our passage retrieval technique was used for query expansion via local feedback. For ranking we used a variant of the Okapi formula [7] — specifically BM25 with $b = 0.6$, $k_1 = 1.5$, $k_2 = 0$ and $k_3 = \infty$. One of these runs (`uwmt8a1`) was based on the topic titles only; the other run (`uwmt8a2`) was based on the title and description.

For each query Q we generated all n-covers for the query for all values of n between 1 and the size of the query $|Q|$. Each n-cover was scored using equation 2 and the passages associated with the best 100 n-covers were used for local feedback.

The local feedback processes discarded those n-covers whose length exceeded a threshold value of 256 words and expanded passages whose length was less than 32 words symmetrically to 32 words. In addition, a term was only considered for local feedback if it appeared in at least two different passages. Since passages may overlap, and the same term position may be part of two different passages, terms were also required to appear in at least two distinct positions. Stopwords were also eliminated from consideration.

	before expansion	after expansion	
title-only	average precision	0.2401	0.2673 (+11.3%)[1]
	precision@5	0.5040	0.5560 (+10.3%)[1]
title+desc	average precision	0.2637	0.2671 (+1.2%)[2]
	precision@5	0.5520	0.5280 (−4.3%)[2]

[1] `uwmt8a1`
[2] `uwmt8a2`

Figure 2: Effects of query expansion.

A score was computed for each term as the sum of the scores of the passages in which it was contained. The top 30 terms were used in the final query.

Figure 2 shows the results of query expansion. We were disappointed in the effects of query expansion. Preliminary experiments with the TREC-7 queries led us to expect improvements of 20% or better. Five other groups achieved a better average precision on their official title-only runs. However, `uwmt8a1` achieved the best precision@5 of all title-only runs.

4.2 Question Answering

We submitted a single run (`uwmt8qa1`) to the Question Answering track. Each question was treated as a query and run using the passage retrieval technique described in Section 4.1. The top five passages were filtered to remove tags and to reduce whitespace, truncated to 250 bytes, and submitted as the "answers".

The run achieved a mean reciprocal rank of 0.471, the sixth highest of the 41 runs.

References

[1] Charles L. A. Clarke and Gordon V. Cormack. Interactive substring retriveal. In *Fifth Text REtrieval Conference (TREC-4)*, pages 295–304, Gaithersburg, Maryland, November 1996.

[2] Charles L. A. Clarke, Gordon V. Cormack, and Forbes J. Burkowski. Shortest substring ranking. In *Fourth Text REtrieval Conference (TREC-4)*, pages 295–304, Gaithersburg, Maryland, November 1995.

[3] Charles L. A. Clarke, Gordon V. Cormack, and Elizabeth A. Tudhope. Relevance ranking for one to three term queries. In *Fifth RIAO Conference*, pages 388–400, Montreal, June 1997. A version of this paper will appear in *Information Processing and Management*, 2000.

[4] Gordon V. Cormack, Charles L. A. Clarke, Christopher R. Palmer, and Samual S. L. To. Passage-based refinement. In *Sixth Text REtrieval Conference (TREC-6)*, pages

303–319, Gaithersburg, Maryland, November 1997. A version of this paper will appear in *Information Processing and Management*, 2000.

[5] Gordon V. Cormack, Christopher R. Palmer, Michael Van Biesbrouck, and Charles L. A. Clarke. Deriving very short queries for high precision and recall. In *Seventh Text REtrieval Conference (TREC-7)*, pages 121–132, Gaithersburg, Maryland, November 1998.

[6] David Hawking, Ellen Voorhees, Nick Craswell, and Peter Bailey. Overview of the TREC-8 Web Track. In *Eighth Text REtrieval Conference (TREC-7)*, Gaithersburg, Maryland, November 1999.

[7] S. E. Robertson, S. Walker, and M. Beaulieu. Okapi at TREC-7. In *Seventh Text REtrieval Conference (TREC-7)*, pages 253–264, Gaithersburg, Maryland, November 1998.