Inf Retrieval (2007) 10:297–319
DOI 10.1007/s10791-007-9023-y

Result merging methods in distributed information retrieval with overlapping databases

Shengli Wu · Sally McClean

Received: 27 August 2004 / Accepted: 2 November 2005 /
Published online: 9 February 2007

Abstract In distributed information retrieval systems, document overlaps occur frequently among different component databases. This paper presents an experimental investigation and evaluation of a group of result merging methods including the shadow document method and the multi-evidence method in the environment of overlapping databases. We assume, with the exception of resultant document lists (either with rankings or scores), no extra information about retrieval servers and text databases is available, which is the usual case for many applications on the Internet and the Web.

The experimental results show that the shadow document method and the multi-evidence method are the two best methods when overlap is high, while Round-robin is the best for low overlap. The experiments also show that [0,1] linear normalization is a better option than linear regression normalization for result merging in a heterogeneous environment.

Keywords Result merging · Distributed information retrieval · Overlapping databases

1 Introduction

With the rapid development of the Internet and WWW, numerous on-line resources are available due to the efforts of research groups, companies, government agencies and individuals all over the globe. How to provide an effective and efficient solution to access such a huge collection of resources for end users is a demanding issue, which is the major goal of distributed information retrieval (DIR) systems. Typically, a DIR system's working process can be divided into several steps as follows:

S. Wu (✉) · S. McClean
School of Computing and Mathematics, University of Ulster, Northern Ireland, UK
e-mail: s.wu1@ulster.ac.uk

S. McClean
e-mail: si.mcclean@ulster.ac.uk

1. For any query submitted from a user, a group of resources needs to be specified either by the user manually or by the DIR system automatically from all available resources for retrieving;
2. The DIR system sends the query to all selected resources. Each resource possesses a text database and a retrieval server, and retrieves a group of documents for the given query. Then, the DIR system collects the results from all these selected resources;
3. Finally the DIR system merges these results into a single list and presents them to the user.[1]

Because many different kinds of search engines can be involved, a DIR system has to deal with different operational problems which are mainly caused by the heterogeneity of different component search engines. Component search engines can be different in many ways from collection coverage, retrieval strategy, document representation and indexing, to query representation. Differences can also originate from result format, response time and retrieval quality. When developing a DIR system, resource description, resource selection and result merging are identified as the three major issues. Among them, high quality of result merging should guarantee that the most relevant documents appear at the top of probably a long list of documents. Such a list is convenient for users so that they may find the most relevant documents in the first one or two pages and do not need to go through the whole result list.

In previous research work, very often it is assumed that textual databases in different resources were totally different from each other. The effect of overlaps among textual databases was not considered in general. However, partial overlaps happen in many situations. For example, there are dozens of general-purpose Web search engines on the Web, their document databases overlap considerably with each other (searchengineshowdown[2]). For a portal or meta-search engine to retrieve documents from them, overlapping should be considered in the result merging for better performance. Even for classical search engines, different portions of the Web may be stored on different computers due to a huge amount of available Web pages that need to be indexed. In this case, the overlap is generated by mirror Web sites and merging is also important. Another example concerns the scientific literature. Every year, scientists and researchers publish a huge number of papers in various periodicals and conferences. Many of these publications are available on-line through digital libraries and/or electronic information services. Considering computer science-related databases, there are dozens of them (e.g. ACM digital library, Applied Science and Technology Plus, Compendex, Computer and Information Systems Abstracts, CSA, Current Contents Connect, DTIC's Scientific & Technical Information Network, Electronics and Communications Abstracts, IEEE/IEE Electronic Library Online—IEEE Xplore, Internet Guide to Engineering, Mathematics, and Computing, Science Citation Index, Web of Knowledge and many others) available. In the earlier two examples, considerable overlaps exist among the databases but they are not identical by any means. To provide a service that can search many of them automatically at the same time is definitely an advantage for use. However, as we will demonstrate in this paper, a solution which is ignorant of overlaps among databases may degrade the quality of retrieval

[1] There are some other options. One is the hierarchical presentation. All documents in the result list are organized into a hierarchy, in which similar documents are clustered together. The other is presenting the results by separated lists, where each list is the original list of documents from a resource; no result merging is needed in this situation. However, the single list solution is the commonest approach, and we only consider this in this paper.

[2] Searchingshowdown is a website that provides information about web search.

considerably. Therefore, the effect of overlapping should be investigated in several activities such as resource selection and result merging in DIR systems.[3]

This paper is focused on result merging methods for overlapping databases. In such a circumstance, we have more information (i.e. the document duplicates) than with disjoint databases. We may use this to improve the effectiveness of the merged result in two different ways. First, duplicate documents in different results may be helpful for correlating the scores (rankings) of all documents in these results. Second, we may use the multiple evidence principle as in data fusion (merging results from identical databases), which regards a document appearing in multiple results as increasing evidence of being relevant to the given query. However, some effective data fusion methods may not be appropriate for overlapping databases, therefore, some adjustment to existing data fusion methods and/or new result merging methods are needed for overlapping databases.

The rest of this paper is organized as follows: in Section 2, we review some previous work on result merging in DIR systems. Section 3 discusses the factors that need to be considered for merging results from overlapping databases, and presents four merging methods. Experimental settings and results for the evaluation of these result merging methods are presented in Section 4. As one important aspect that affects result merging, Section 5 presents experimental results for the tendency of overlap to exist in the case of two overlapping databases for the same query. Section 6 concludes the paper.

2 Related work

In a survey paper, Meng et al. (2002) identified three cases for result merging, which depend on the degree of overlap among the selected databases for a given query:

1. the databases are disjoint or nearly disjoint;
2. the databases are overlapping but are not identical;
3. the databases are identical.

Cases 1 and 2 may occur quite often in DIR environments. For example, a group of special-purpose Web search engines or digital libraries on different subjects may have very little overlap among their databases, while several general-purpose Web search engines or digital libraries on the same subject may have considerable overlap among their databases. Case 3 is an ideal situation and usually does not occur in a typical DIR environment. However, it could be exploited to improve retrieval effectiveness in certain situations. The result merging problem in this case is also known as data fusion.

There has been a lot of research on data fusion. We only review a few references which are related to this paper. Fox and Shaw (1994) proposed a number of combination techniques including operators like Min, Max, Ave, CombSum and CombMNZ. CombSum determines the score of each document in the combination as the sum of the scores obtained by the individual resources, while in CombMNZ the score of each document is obtained by multiplying this sum by the number of resources which have non-zero scores. Note that summing (CombSum) is equivalent to averaging, while CombMNZ is equivalent to weighted averaging.

Lee (1997) studied this issue further with six different servers. His contribution was to normalize each information retrieval server on a per-query basis which improved results

[3] How to detect (partially) overlapping documents is a relevant issue. Some research on this has been published, for example, in Chowdhury et al. (2002).

substantially. Lee showed that CombMNZ worked best, followed by CombSum, while operators like Min and Max were the worst.

Linear combination was described by Vogt and Cottrell (1999). In this approach, the relevance of a document to a query is computed by combining both a score that captures the quality of each resource and a score that captures the quality of the document with respect to the query. Formally, if q is a query, d is a document, n is the number of resources, $w = (w_1, w_2, \ldots, w_n)$ are the resource scores, and $s = (s_1(d, q), s_2(d, q), \ldots, s_n(d, q))$ are the scores of document d in all resources. Then, the overall score $s(w, d, q)$ of d in the context of the combined list is given by

$$s(w, d, q) = \sum_{i=1}^{n} w_i s_i(d, q)$$

Aslam and Montague proposed several data fusion methods using Bayesian inference, Borda fusion and Condorcet fusion (Aslam and Montague, 2003; Montague and Aslam, 2002). These methods are close to CombMNZ in performance.

Score distribution models for relevant and non-relevant documents were proposed by Manmatha et al. (2001). These models could be useful for score normalization which is needed in all data fusion methods.

For non-overlapping databases, Round-robin is a simple merging method, which takes one document, in turn, from each available result list. However, the effectiveness of such a method depends on both the performances of component resource servers and the contents of component databases. If all the results are of similar quality, then Round-robin performs well; if some of the results are very poor, then Round-robin becomes quite poor as well.

Voorhees et al. (1995) demonstrated a way of improving the earlier-referred Round-robin method. By running some training queries, they estimated the performance of each resource. When a new query was encountered, they retrieved a certain number of documents from each resource based on its estimated performance.

Callan (2000) and Callan et al. (1995) proposed a merging strategy based on the scores achieved by both resource and document. The score for each resource is calculated by a resource selection algorithm, CORI, which indicates the "goodness" of a resource with respect to a given query among a set of available resources. The score of a document is the value that document obtains from the resource, which indicates the "goodness" of that document for the given query among a set of retrieved documents from that resource. Yuwono and Lee (1997) proposed another method for converting local document scores to global scores.

Rasolofo et al. (2003) reported their study on a current news meta-searcher. They found that a low-cost merging scheme based on a combination of available evidence (title, summary, rank and resource usefulness) worked almost as well as merging based on downloading and re-scoring the actual news articles.

Resource selection is another important issue for DIR systems. Such a process is necessary when the available resources are so many that it is only possible to select a subset of them for retrieval. Resource selection has been investigated by many researchers, for example, in Callan (2000), Fuhr (1999), Gravano et al. (1999), Hawking and Thistlewaite (1999) and Wu and Crestani (2003) for various kinds of DIR systems. The ranking of resources obtained at this stage is also useful in data fusion for the estimation of the quality of component results.

Finally, let us see a few result merging methods used by meta-search engines, for which the technical details are available.

MetaCrawler (Selberg and Etzioni, 1995, 1997) treated all databases equally, normalizing scores into the same range between 0 and 1000, then using the Sum function for result merging. Profusion (Gauch et al., 1996) used a method, which was similar to CORI, to calculate a ranking score for each database, and used that to adjust the local score of any document into a global score. The Max function was then used for result merging. Note that Sum and Max are equivalent to Fox and Shaw's CombSum and Max (Fox and Shaw, 1994), respectively. Inquirus (Lawrence and Giles, 1998a,b) took the approach of downloading all the documents. All the documents were re-ranked by analysing the contents of the downloaded documents. A similar approach was taken by Mearf (Oztekin et al., 2002). Some further issues that need to be considered for implementing a meta-search engine are discussed, for example, in Hawking and Robertson (2003) and Lawrence and Giles (2000) and Selberg and Etzioni (2000).

3 Merging results from overlapping databases

For overlapping databases, especially when the overlap is considerable, it is not a good solution for us to use methods proposed for pair-wise disjoint databases, since in the former case, we have more information (the document duplicates) than that in the latter case. We may use this information in two different ways. First, the duplicate documents in different results may be helpful for correlating the scores of all the documents contained in different results. As in Calvé and Savoy (2000) and Si and Callan (2002), regression analysis is a useful tool for achieving this. We can do this as long as there are overlaps among databases, no matter what the overlap rate is. However, we can still use the multiple evidence principle as in data fusion to improve the performance of result merging. However, two things should be noted. First, the multiple evidence principle can only be effectively applied when the overlap among databases is considerable; second, we may need different result merging methods for overlapping databases from that used in data fusion. Let us illustrate this by an example. Suppose one document d_1 is retrieved from database A, and another document d_2 is retrieved from both databases A and B, if we know that A and B are identical, which is the situation for data fusion, then we are sure that d_1 is stored in B but not retrieved by B's server. If A and B are not identical, there are two possibilities: either d_1 is stored in B but not retrieved, or d_1 is not stored in B at all. However, we do not know exactly which situation has occurred.

Based on different assumptions that could be made, we may have different kinds of solutions. For any document d, which occurs in database A's result but not in B's result, if we always assume that d is in B but not retrieved by B's server (*Assumption 1*), then such an assumption is appropriate for identical databases or nearly identical databases. Actually, all data fusion methods with identical databases rely on this since the assumption is always true.

For overlapping databases, especially when the overlap is not very heavy, this assumption is error-prone. We may here make another assumption (*Assumption 2*). That is, if d is not retrieved by B's server, we simply assume it is not stored in B.

The third option is dynamic estimation. We may make one of the earlier assumptions depending on certain circumstances. For better estimation, the overlap rate between two databases is useful information. However, usually it is quite difficult and very costly to obtain accurate information about it in a DIR environment, since all component databases are always under change, and on the contrary it is not common that database servers may provide interfaces to the public for accessing their databases directly. Therefore, in this paper, we assume that no extra information is available except the results themselves.

All data fusion methods can be regarded as making Assumption 1, while all result merging methods investigated in this paper take Assumption 2. As experiments in Wu and Crestani

(2004) and later in this paper suggest: Assumption 2 is more suitable than Assumption 1 for overlapping databases, even when the overlap is quite high. The reason for that is based on the following hypothesis:

For merging results from overlapping databases, when we make Assumption 1, the benefit we obtain from a correct estimation is less than the damage we sustain from a wrong estimation.

This hypothesis can explain why some data fusion methods such as CombMNZ deteriorate very quickly when the databases gradually deviate from being identical. As we shall see later in this paper, CombMNZ performs quite poorly even when the overlap is very high (e.g. as defined in Section 4.2, the overlap rate among databases is up to 80%).

When we make Assumption 2, one thing we can do for a more accurate estimation is to retrieve more documents in every component database. For example, suppose the DIR system needs to return 100 documents to the user for a given query, and four component databases are available, then usually retrieving 30–40 documents from each database is enough. However, if we retrieve more documents from each database, then we know with more certainty whether the top-ranked documents of each database are stored in other databases or not. In Section 4, we shall discuss this in more detail.

Another problem we face is how to merge documents which are retrieved by a different number of databases. It is a new situation that does not happen in data fusion with identical databases. For example, d_1 is only stored in A, d_2 is stored in both B and C; for a given query, d_1 is retrieved by A with a score of 0.6, d_2 is retrieved by B with a score of 0.4 and C with a score of 0.4. How to rank these two documents is a key issue. In the following, we discuss four solutions for this: the shadow document method (SDM), the multi-evidence method (MEM), the Borda fusion method and the Bayesian fusion method.

SDM and MEM assume that every document retrieved from any available database has a score which measures the estimated relevance of that document to the given query, Round-robin and Borda fusion require that in every result all documents are ranked, while Bayesian fusion can work in either way: scores or rankings.

3.1 The shadow document method (SDM)

The shadow document method (SDM) (Wu and Crestani, 2004) works like this. For a given query q and two component databases A and B, if document d occurs in both A and B's results then we sum these two scores as d's global score; if d only occurs in A's result with a score s_1, then it is not stored in B (according to Assumption 2); and we imagine if d is stored in B, it is very likely to be retrieved with a score close to s_1 (normalized). For the shadow document of d in B, we assign a score to that (shadow) document of d in B. We do the same for those documents which only occurs in B's result but not in A's result. Then, we are able to use the CombSum method to merge all the results. Generally, if we have a query q and n databases D_i $(1 \leq i \leq n)$ for retrieving, and document d occurs in m databases with score s_i $(1 \leq i \leq m \leq n)$, respectively, then d's total score is:

$$score(d) = \sum_{i=1}^{m} s_i + k\frac{n-m}{m}\sum_{i=1}^{m} s_i \tag{1}$$

where k is a weighting coefficient. For each result without d, we assign it a value that is the average score of d from m databases multiplied by k. To assign a desirable value to k is important and it needs to be determined empirically.

Besides, we can estimate the performance of each component resource by some sample queries or based on previous retrieval experience, then each resource obtains a score corresponding to its estimated performance. The earlier-referred shadow document method can be improved by this resource score as in linear combination. However, we do not discuss weighted result merging methods further in this paper, since most methods discussed in this paper can be expanded with weighting straightforwardly.

3.2 The multi-evidence method (MEM)

In this method, we first average the score of every document, then multiply that score by a factor $f(i)$ that is a function of the number of databases that include the document in their results. The value of $f(i)$ increases with i, which indicates the increasing evidence about the relevancy of the document to the given query when more parties have a common opinion. For example, if d_1 is retrieved by A with a (normalized) score of 0.6, and retrieved by B with a (normalized) score of 0.5, then the combined score is $f(2) * (0.6 + 0.5)/2$. How to determine $f(i)$ for $(i = 1, 2, 3, \ldots)$ is important to this method.

If we use a different $f(i)$, we may obtain a very different result merging method. For example, if we let $f(i) = 1$ for $(i = 1, 2, 3, \ldots)$, then the multi-evidence method is just equal to CombSum, and no multi-evidence is considered. If we let $f(i) = i$ for $(i = 1, 2, 3, \ldots)$, then we have CombSum; if we let $f(i) = i^2$ for $(i = 1, 2, 3, \ldots)$, then we have CombMNZ.

Lee (1997) did an experiment for data fusion with six text retrieval engines. He let $f(i) = avg\_score * i^{\beta}$ with different β values (1, 1.5, 2, 3, 6, and 11) assigned to $f(i)$. He found that it worked best when $\beta = 2$, which was exactly the CombMNZ method. This suggests that with identical databases, we should set a weight as heavy as i^2 for those documents retrieved by multiple resources. However, for overlapping databases, the situation is different. $f(i)$ should be defined with a value between 0 and 2 for β.

A more sophisticated solution is to let $f(i)$ vary with the degree of overlap among databases. If the overlap is high, we may use a bigger β that approaches 2; if the overlap is low, we may use a smaller β. However, in this paper we only consider static $f(i)$ which is used in all different situations. We leave the adaptable $f(i)$ issue for further investigation.

3.3 The Borda fusion method

The Borda count was initially proposed in political science for democratic voting. It can be applicable for data fusion in which there are many candidates (retrieved documents) and relatively few voters (information retrieval systems). It has recently been shown that the Borda count is optimal in the sense that only the Borda count satisfies all of the symmetry properties that one would expect of any reasonable election strategy.

The Borda count in data fusion (referred to as Borda fusion, Aslam and Montague (2003) works as follows. Each voter ranks a fixed set of c candidates in order of preference. For each voter, the top-ranked candidate is given c points, the second-ranked candidate is given $c - 1$ points and so on. If there are some candidates left unranked by the voter, the remaining points are divided evenly among the unranked candidates. The candidates are ranked in order of total points, and the candidate with the most points wins the election. In such a way, every ranked document obtains a point, and we may use the same method as in the shadow document method (see Eq. ((1))) for overlapping databases.

3.4 The Bayesian fusion method

Bayesian fusion (Aslam and Montague, 2003) is based on Bayesian inference. Given the ranked lists of documents returned by n retrieval systems, let $r_i(d)$ be the rank assigned to document d by retrieval system i. This constitutes the evidence of relevance provided to the merging strategy concerning document d. For a given document, let $P_{rel} = Pr[rel|r_1, r_2, \ldots, r_n]$ and $P_{\text{irr}} = Pr[irr|r_1, r_2, \ldots, r_n]$ be the respective probabilities that the given document is relevant and irrelevant given the rank evidence $r_1, r_2, \ldots, r_n$. We compute the log odds of relevance: $\log O[rel] = \log P_{rel}/P_{irr}$ and rank documents according to this measure. According to Hull et al. (1996), we have

$$\log O[rel|r_1, r_2, \ldots, r_n] = \sum_{i=1}^{n} \log O[rel|r_i] - (n-1)\log O[rel].$$

This formula can be proven by induction as follows (Montague, 2002). The base case is trivial. Now if we assume that the formula holds for $n-1$, we can show that it holds for n.

$$\begin{aligned} O[rel|r_1 \ldots r_n] &= \frac{Pr[rel|r_1 \ldots r_n]}{Pr[irr|r_1 \ldots r_n]} \\ &= \frac{Pr[r_1 \ldots r_n|rel]Pr[rel]}{Pr[r_1 \ldots r_n|irr]Pr[irr]} \\ &= \frac{Pr[r_1 \ldots r_{n-1}|rel]Pr[r_n|rel]Pr[rel]}{Pr[r_1 \ldots r_{n-1}|irr]Pr[r_n|irr]Pr[irr]} \end{aligned}$$

(independence is assumed in these equations)

$$\begin{aligned} &= \frac{Pr[rel|r_1 \ldots r_{n-1}]Pr[rel|r_n]Pr[irr]}{Pr[irr|r_1 \ldots r_{n-1}]Pr[irr|r_n]Pr[rel]} \\ &= \frac{O[rel|r_1 \ldots r_{n-1}]O[rel|r_n]}{O[rel]} \end{aligned}$$

So,

$$\log O[rel|r_1 \ldots r_n] = \log O[rel|r_1 \ldots r_{n-1}] + \log O[rel|r_n] - \log O[rel].$$

Applying our inductive hypothesis yields the desired result.

Dropping terms that are the same for all documents gives a formula for calculating the log odds of relevance for ranking:

$$score(d) = \sum_{i=1}^{n} \log O[rel|r_i].$$

Note that either rankings or scores can be used in this formula. For better estimation, training is required for parameters setting. Finally, we may use the same method as in the shadow document method (see Eq. (1)) for overlapping databases.

4 Evaluation of result merging methods

Extensive experimentation has been done to evaluate the performances of a group of result merging methods. Here, we assume that no extra information is available except the result lists from different resources. Especially, we do not know if any particular document is stored in any particular database, and we do not know any statistical information about the sizes of databases and overlapping rate of these databases, and so on. Besides, we assume that every retrieval server performs equally well.

In the experiment, different available information such as scores or rankings of the documents in the result and different normalization methods of scores are considered. In all these methods, both Round-robin and Borda use rankings, and all other methods use scores. For result merging, normalization of scores is an important step. [0,1] linear normalization and linear regression normalization have been used among others. We aim to observe the difference caused by these two different normalization methods for result merging.

4.1 The overlap rate among databases

We use *overlap_rate* to measure the degree of overlap among a group of databases. It is defined as follows:

$$overlap\_rate = \frac{(\sum_{i=1}^{n} |D_i|) - |D_{all}|}{(n-1) * |D_{all}|}$$

where n is the number of databases involved, $|D_i|$ is the number of documents in database D_i and $|D_{all}|$ is the total number of different documents in all n databases.

When there are no duplicates, $\sum_i^n |D_i| = |D_{all}|$, $overlap\_rate = 0$; when all the databases have the same documents, $|D_i| = |D_{all}|$ for $(i = 1, 2, \ldots, n)$, then $overlap\_rate = 100\%$.

For example, if we have 100,000 different documents in total, database D_1 includes 70,000 documents, D_2 includes 50,000 documents and D_3 include 30,000 documents, then $overlap\_rate = \frac{70000+50000+30000-100000}{2*100000} = 25\%$.

4.2 Experimental settings

TREC text collections and topics were used. 50 TREC ah hoc topics (251–300) and 524,929 documents (AP88, CR93, FR88, FR94, FT91-94, WSJ90-92 and ZF, which comprised the collection used by the TREC 5 retrieval tasks (Voorhees and Harman, 1996)) were used.

Both Lemur[4] and Lucene[5] information retrieval tool kits were used. Lemur provides three options as retrieval models: vector space, okapi and the language model, while Lucene provides two options, the vector space and the probabilistic model,[6] as retrieval models.

[4] Lemur is an open-source toolkit designed to facilitate research in language modeling and information retrieval. Lemur was implemented by researchers in Carnegie Mellon University and University of Massachusetts.

[5] Lucene is another free and open source information retrieval API, originally implemented in Java by Doug Cutting. It is supported by the Apache Software Foundation and is released under the Apache Software License.

[6] The probabilistic model (see pp. 30–34 in Baeza-Yates and Ribeiro-Neto (1999)) was implemented by one of the authors of this paper.

All the documents were divided into 250 partitions. Each included around 2,000 documents. The generation of five databases is as follows: each partition is included in at least one database, then it is guaranteed that the number of different documents in all five databases is invariant. We used a random process to determine which database included which partition and how many databases included any given partition. An adjustable parameter *overlap_rate* was used to help control the overlapping rate among five databases. *overlap_rate* takes value in five intervals *A*: (0.0, 0.2), *B*: (0.2, 0.4), *C*: (0.4, 0.6), *D*: (0.6, 0.8) and *E*: (0.8, 1.0). Three different values are used for each interval. We used 0.05, 0.15, 0.25, 0.35, 0.45, 0.55, 0.65, 0.75, 0.85 and 0.95 three times, and 0.10, 0.20, 0.30, 0.40, 0.50, 0.60, 0.70, 0.80 and 0.90 four times. Then, 10 values were taken in every interval. For every partition, the documents are randomly assigned to one of the five databases, then the probability of assigning the documents to any other database is determined by *overlap_rate*.

After generation, the documents were indexed and retrieved with the same TREC queries. Each database used a different retrieval model in either Lemur or Lucene. We consider this is more likely to be the case for real applications, especially in loosely coupled DIR systems. Then, their results were merged using different merging methods.

In the experiment, six methods namely Round-robin, Borda fusion, Bayesian fusion, MNZ (CombMNZ), SDM and MEM, were tested. Round-robin served as the baseline.

As indicated in Section 3, some of the methods need a few parameters. For Bayesian fusion, we first normalize scores of documents from one retrieval server into the range [0, 0.5]. Here 0.5 is chosen as the probability of the top-ranked documents, because on average, about half of the top-ranked documents are relevant to the queries. In such a way, since all scores calculated in Bayesian fusion are negative (they are the log of terms less than 1), parameter k needs to be set no less than 1. Twenty-one different values (1, 1.05, 1.1, ... , 3) were tested in the experiment. For both Borda and SDM, $k = 0.0, 0.05, 0.1, \ldots, 1$ was tested. For MEM, we let $f(i) = 1 + \log(i)$.

For a fair comparison of the performance of several result merging methods, it is necessary to set these parameters in a way that does not favour any particular method. For each parameter, we choose a typical value for it. $k = 0.5$ is chosen for SDM, $k = 0.95$ for Borda fusion, and $k = 2.5$ for Bayesian fusion. These values are chosen so as to provide reasonable, but not necessarily optimal, result merging. In addition, the effect of k on the performance of two methods (SDM and Bayesian fusion) is analysed in detail later in this section. The results presented in Section 4.3 confirm that these choices for parameters setting are reasonable.

4.3 Experimental results

We used a different model in either Lemur or Lucene to retrieve each database in a quintuple. Different result merging methods were evaluated by using Round-robin as baseline. We divide the possible value range [0,1] of *overlap_rate* into five equal parts, each table presents the result with *overlap_rate* locating in one part. Each item in these tables is the average of 10 quintuples over 50 TREC ad hoc topics. Therefore, in each set of 10 tables, the first five of them use [0,1] linear normalization, while the last five use linear regression normalization (see Appendix).

Because all five retrieval models in Lemur and Lucene used in the experiment are quite good and close in performance, the relevant documents are evenly distributed in each of the five databases, and the documents in all quintuples are the same, the performance of Round-robin is very stable in all different overlapping situations. Compared with Round-robin, all other methods do better in the top 5 or 10 ranked documents. CombMNZ deteriorates more quickly and further in performance than any other method when the overlap rate falls. This

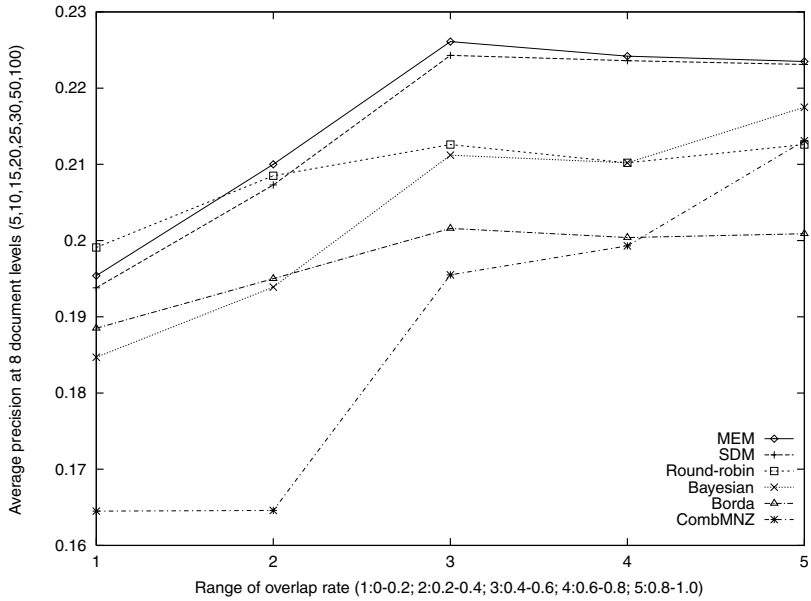

Fig. 1 Performances of six methods with different overlap rates ([0,1] normalization)

is not surprising since all these methods exploit the multiple evidence principle to boost the rankings of those commonly retrieved documents, while Round-robin does not. Because CombMNZ relies most on this principle, it suffers the most when the overlap rate is low. In the same situation, SDM, MEM, Borda fusion and Bayesian fusion do not suffer so much since a balanced mechanism are introduced.

In the following, let us discuss the experimental result in two groups, each with a different normalization method. First with [0,1] normalization, Fig. 1 shows the average precision of six methods at eight document levels (5, 10, 15, 20, 25, 30, 50 and 100) with different overlap rates. Both SDM and MEM are close in performance in most situations. They perform better than Round-robin when the overlap rate among databases is no less than 40% (the differences are not always significant), they are close to Round-robin when the overlap rate is between 20% and 40% and they become not as good as Round-robin when the overlap rate is less than 20% (the differences are not significant in most cases). CombMNZ is not as good as SDM and MEM in all cases (the maximal overlap rate used is about 95% in the experiment). However, we do not try to find out in which condition CombMNZ is better than SDM and MEM. If it exists, it must be very close to 100%. Bayesian fusion does not work as well as SDM and MEM. Its performance is slightly better than Round-robin when the overlap rate is more than 80%. Borda fusion is not as good as Round-robin in all situations.

Figure 2 shows the average precision of four methods at eight document levels (5, 10, 15, 20, 25, 30, 50 and 100) with linear regression. SDM and MEM perform slightly better than Round-robin when the overlap rate is above 80%. When the overlap rate is between 60% and 80%, the performances of SDM and Round-robin are close. CombMNZ's performance is close to Round-robin's when the overlap rate is over 80%. In all other situations, Round-robin outperforms all three other methods.

In summary, both SDM and MEM perform better with [0,1] linear normalization than with linear regression normalization in most situations and on average. The only exception

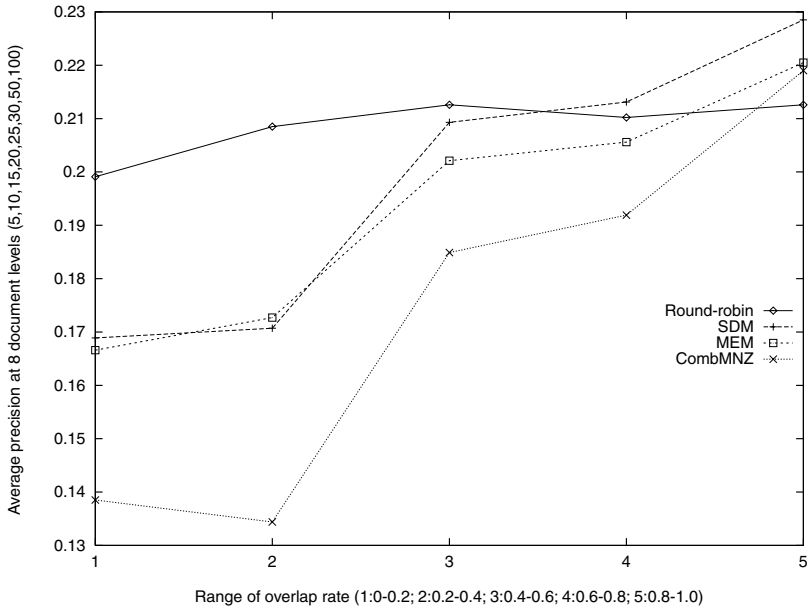

Fig. 2 Performances of four methods with different overlap rates (regression normalization)

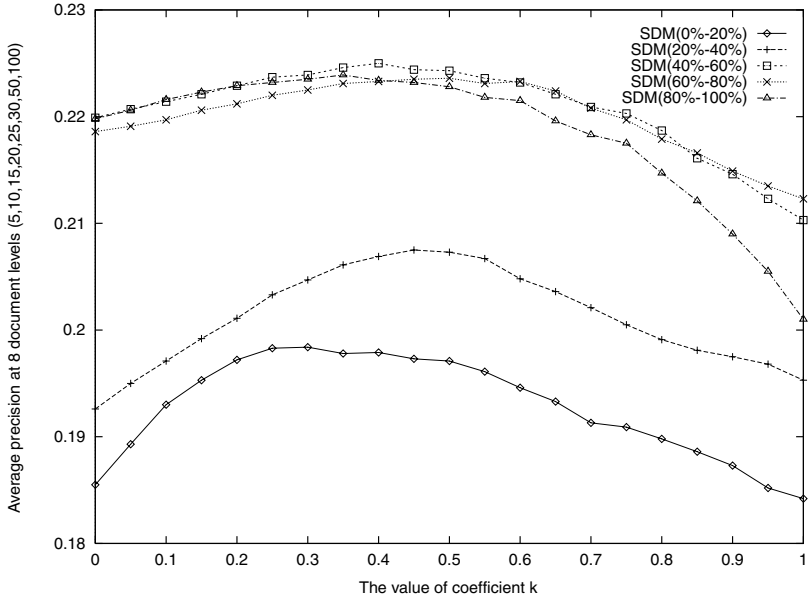

Fig. 3 SDM's average performance with different coefficient values

we observe is SDM with overlap rates between 80 and 100%. Besides, using [0,1] linear normalization, all result merging methods are more stable in performance under different overlap rates than using linear regression normalization. Actually, such a phenomenon is quite

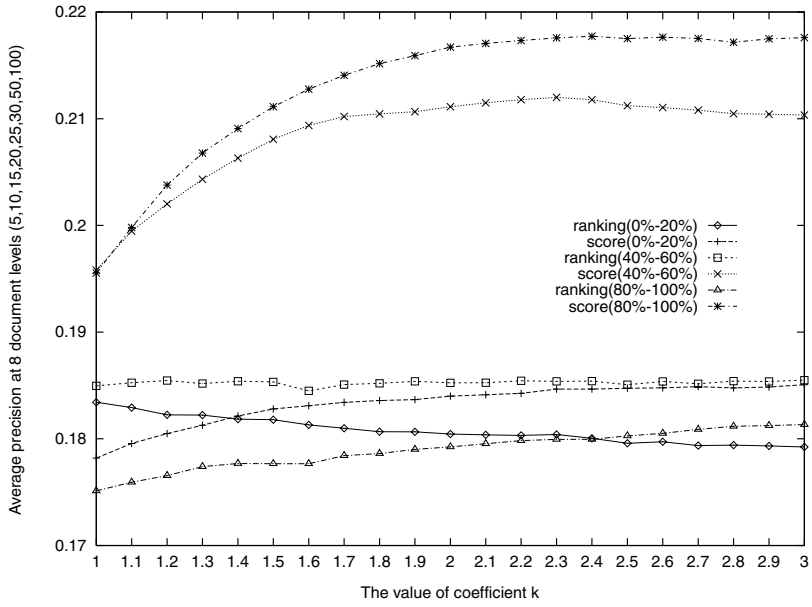

Fig. 4 Comparison of using score and ranking in Bayesian fusion

surprising, since in Wu and Crestani (2004), linear regression normalization was successfully used with the SDM method. The only observable difference is that only Lemur was used in Wu and Crestani (2004), while both Lemur and Lucene have been used here. This suggests that [0,1] linear normalization is a better option than linear regression normalization for result merging in a heterogeneous environment.

Figure 3 shows the average precision at eight document levels (5, 10, 15, 20, 25, 30, 50 and 100) of SDM ([0,1] linear normalization) with different overlap rate and different coefficient values of k. We can observe that SDM's performance is the lowest when the overlap rate is between 0% and 20%, then it is better when the overlap rate increases to between 20% and 40%, and it is the best when the overlap rate is above 40%. However, when the overlap rate is no less than 40%, further increase in the overlap rate is not useful for improving retrieval quality. Please note the three curves for SDM(40–60%), SDM(60–80%) and SDM(80–100%), they are very close to each other or even converging. We observe that the same thing happens to MEM, Borda fusion and Bayesian fusion with different overlap rates (see also Fig. 1).

In addition, Fig. 3 shows the effect of different k on the performance of SDM. This suggests that a value between 0.2 and 0.6 for k is acceptable.

We experimented with Bayesian fusion using score and ranking, respectively (figures shown in Tables 1–10 using scores). Figure 4 shows the average precision of Bayesian fusion with three different overlap rates (0–20%, 40–60%, 80–100%). With ranking information only, all three curves are very close; while a significant difference can be observed for three curves using score information. The "score(0–20%)" is the lowest, the "score(40–60%)" is in the middle, while the "score(80–100%)" is the highest. Let us look at them in three pairs, each with the same overlap rate but different available information (score or ranking). For the pair with overlap rate between 0 and 20%, the curves are quite close to each other. For the two other pairs (overlap rate between 40 and 60% and between 80 and 100%), the

difference between the two curves is quite big (10–15%). Since all other aspects are identical, it confirms that score is more informative than ranking in result merging.

5 Overlap between different results

We hypothesize that all "good" text retrieval servers are very similar from a user's point of view. That is to say, all "good" text retrieval servers retrieve a considerable number of identical documents for the same query. One possible solution to addressing this issue is to analyse all text retrieval servers involved. However, this is not a very effective approach. For example, it is not clear how different it is between a probabilistic model and a vector space model. Also, it is not clear how much difference a query expansion component can bring to a text retrieval server. However, to compare the results from different retrieval servers for the same information need is a practicable solution. This is the approach that we take in this paper.

Generally speaking, the number of duplicates in two different results may depend on several aspects:

1. the number of documents in each result;
2. the size and content of each database;
3. the size of the overlap between the two databases;
4. information retrieval mechanisms used in the two databases.

In this paper, we do not intend to study the effect of all these aspects but only a typical situation. An experiment was conducted like this:

1. Create two overlapping databases, in which C is the common document set;
2. Use two different retrieval models and the same query to retrieve both databases so as to get two different results R_a and R_b;

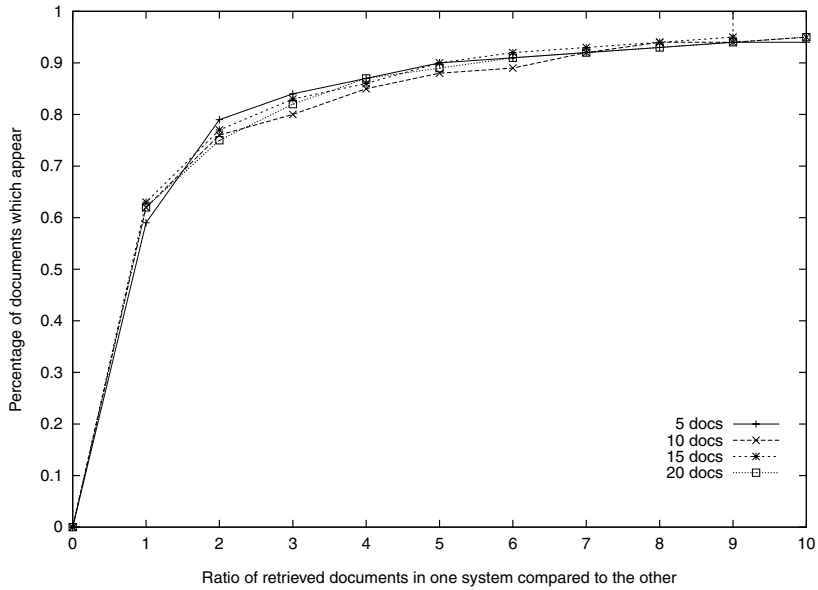

Fig. 5 Overlap between two results for the same query

3. Pick up those documents in the top n of R_a which belong to C, and check how many of them appear in the top $n, 2n, 3n, \ldots$ documents of R_b;
4. Change the positions of R_a and R_b and repeat the last step.

Figure 5 shows the overlap between two retrieved results, which is based on the average of 50 queries over a pair of overlapping databases. Two different retrieval models of Lemur were used for the retrieval of two databases. We took the top n ($n = 5$, 10, 15 and 20) documents in one result, picked up those documents from the overlapping part, then checked how many of them appear in the top $n, 2n, \ldots, 10n$ in the other result, these percentages are presented in the plot, in which the four curves overlap considerably.

For documents which come from the overlapping part that are located in the top n ($n = 5$, 10, 15 or 20) in one result, about 60% of the documents appear in the top n and over 90% of the documents appear in the top $10n$ in the other result. Besides, when 12 more pairs of overlapping databases with different overlap rates and contents were tested, we obtained very similar results as in Fig. 5 in all cases. When we used the same model in Lemur to retrieve these database pairs, then the percentages were higher than that shown in Fig. 5.

From this experiment, we observe that if we require n documents for the merged results, then to retrieve more documents (for example, $8n-10n$ documents) is a good policy which enable us to estimate more accurately if a document exists in a particular database or not. Therefore, retrieving more documents from each component database can improve the performance of result merging methods accordingly.

6 Conclusions

In this paper, we have discussed several result merging methods in distributed information retrieval environments. Extensive experimentation has been carried out with TREC document collections and diverse overlaps among databases to evaluate the performance of these result merging methods to compare average precision at different document levels. Six result merging methods, SDM, MEM, Bayesian fusion, Borda fusion, CombMNZ and Round-robin (serving as baseline), have been evaluated. Two score normalization methods, which are [0,1] linear normalization and linear regression normalization, and two different kinds of available information, which are scores and rankings, have been tested. From these experiments, we have several observations as follows:

1. Experimental results show that CombMNZ, SDM and MEM perform better with [0,1] linear normalization than with linear regression normalization;
2. If we only consider the top 10 documents, then all five methods (CombMNZ, Borda fusion, Bayesian fusion, SDM and MEM) perform better than Round-robin when overlap rate is above 60%; for SDM and MEM, the above requirement can be weakened to 40%. Even when the overlap rate is below 40%, the performances of both SDM and MEM are very close to that of Round-robin.
3. By comparing the performance of Bayesian fusion with two kinds of information, score and ranking, we have demonstrated that score is more informative than ranking for result merging;
4. The experiment also shows that both SDM and MEM are effective methods in most situations. They perform better than Round-robin when the overlap rate is no less than 40%.
5. Borda fusion is not as good as Round-robin in all cases;

6. Both Bayesian fusion and CombMNZ are not as good as Round-robin when the overlap rate is below 80%. When the overlap rate is above 80%, CombMNZ and Round-robin become comparable, while Bayesian fusion is slightly better than Round-robin;
7. Round-robin is quite good in all situations since the retrieval servers used in the experiment are quite close in performance and relevant documents are evenly distributed in every database.

In order to achieve a better performance for result merging methods such as the shadow document method, multi-evidence method and Bayesian fusion, more documents than that appearing in the merged result are required from each component database. For example, if 20 documents are needed in the final result, then retrieving about 200 documents from each component database may lead to a more effective merging than using fewer documents for them.

Some methods such as CombMNZ that are very good for data fusion with identical databases are very poor for result merging with overlapping databases. Even when the overlap is quite high (e.g. when the overlap rate among databases is up to 60%), CombMNZ's performance is not at all good. However, methods such as CombMNZ are not suitable for overlapping databases since they only pick up those documents stored in all or most of the databases but ignore those documents stored in only one or very few databases. Therefore, such merging methods are not suitable on the aspect of coverage.

Some of our on-going research suggests that the strength of correlation among component results affects data fusion (result merging) considerably. We consider the experimental environment set in this experiment as modestly heterogeneous—two different retrieval systems, one with three different models, and the other with two different models. Two other situations, homogeneous and extremely heterogeneous environments, are also worth discussion. In addition, test with some other kinds of collections, for example, in a language other than English, or on a different topic other than news articles, may be helpful for us to understand more about the result merging issue.

One limitation of the current research is: the experimental setting in this paper is similar to the scenario of a distributed digital library system, but different from the scenario of Web search, which is a potential application area of this research. The Web search queries and search engines are quite different from the queries and retrieval systems used in our experiment, and especially the size of Web text document collection is much bigger than the text document collections used in our experiment. Also, we note that the TREC ad hoc evaluation, which assumes that all relevant documents are of equal value and rewards systems which return the most relevant documents, does not model typical Web search. Even so, we consider that the experimental result in this paper is still useful for Web search for two reasons. First, a result from Web search possesses the same characteristic as a result from conventional text retrieval: the top ranked-documents have the highest probability of relevance to the information need, and then the probability decreases when we retrieve more and more documents. Second, all conventional text retrieval systems use statistical techniques, while almost all Web search engines use statistical techniques plus link analysis. Therefore, the similarity among all Web search engines could be comparable to that of all conventional text retrieval systems. However, since it is not possible to carry out such experiments, which require documents to be overlapping with diverse overlap rates, on the Web itself, we need to carry out experiments in a scenario more like the Web for a better understanding of our result merging methods. We plan to carry out such a study in future work. This will involve using Web documents and Web search engines as well as addressing scalability issues.

Appendix

In the following tables, a figure in italic indicates the best performance in that case; and figures with "$*$" indicate that the figure is statistically different from the one in Round-robin at 95% significance level (paired-samples T test).

Table 1 Average precision of different methods with overlap rate between 0 and 20% ([0,1] linear normalization and Round-robin serves as baseline)

Docs rank	Round-robin	MNZ	Bayesian ($k = 2.5$)	Borda ($k = 0.95$)	SDM ($k = 0.5$)	MEM
5	*0.2676*	0.2436* (−9.0%)	0.2336* (−12.7%)	0.2448* (−8.5%)	0.2628 (−1.8%)	0.2648 (−1.0%)
10	*0.2437*	0.2050* (−15.9%)	0.2232* (−8.4%)	0.2280* (−6.4%)	0.2362 (−3.1%)	0.2436 (±0.0%)
15	*0.2228*	0.1860* (−16.5%)	0.2081* (−6.6%)	0.2131* (−4.4%)	0.2173 (−2.5%)	0.2197 (−1.4%)
20	*0.2079*	0.1681* (−19.1%)	0.1962* (−5.6%)	0.1987* (−4.4%)	0.2021 (−2.8%)	0.2023 (−2.7%)
25	*0.1953*	0.1573* (−19.5%)	0.1852* (−5.2%)	0.1852* (−5.2%)	0.1883 (−3.6%)	0.1882* (−3.6%)
30	*0.1830*	0.1464* (−20.0%)	0.1724* (−5.8%)	0.1764* (−3.6%)	0.1789 (−2.2%)	0.1798 (−1.7%)
50	*0.1542*	0.1198* (−22.3%)	0.1466* (−4.9%)	0.1488* (−3.5%)	0.1520 (−1.4%)	0.1516* (−1.7%)
100	*0.1181*	0.0900* (−23.8%)	0.1123* (−4.9%)	0.1127* (−4.6%)	0.1129* (−4.4%)	0.1136* (−3.8%)

Table 2 Average precision of different methods with overlap rate between 20 and 40% ([0,1] linear normalization and Round-robin serves as baseline)

Docs rank	Round-robin	MNZ	Bayesian ($k = 2.5$)	Borda ($k = 0.95$)	SDM ($k = 0.5$)	MEM
5	0.2844	0.2324* (−18.3%)	0.2614* (−21.9%)	0.2502* (−8.1%)	0.2862 (+0.6%)	*0.2916* (+2.5%)
10	0.2584	0.2004* (−22.4%)	0.2392* (−7.4%)	0.2372* (−8.2%)	0.2558 (−1.0%)	*0.2587* (+0.1%)
15	0.2352	0.1780* (−24.3%)	0.2162* (−8.1%)	0.2187* (−7.1%)	0.2338 (−0.6%)	*0.2362* (w + 0.4%)
20	0.2169	0.1632* (−24.8%)	0.2017* (−7.0%)	0.2074* (−4.3%)	0.2158 (−0.5%)	*0.2194* (+0.1%)
25	0.2015	0.1519* (−24.6%)	0.1897* (−5.6%)	0.1950* (−3.2%)	0.2012 (−0.1%)	*0.2026* (+0.5%)
30	0.1899	0.1519* (−20.0%)	0.1788* (−5.8%)	0.1833* (−3.5%)	0.1888 (−0.6%)	*0.1917* (+0.9%)
50	0.1596	0.1438* (−9.9%)	0.1505* (−5.7%)	0.1525* (−4.4%)	0.1588 (−0.5%)	*0.1597* (±0.0%)
100	*0.1224*	0.0948* (−22.5%)	0.1138* (−7.0%)	0.1159* (−5.3%)	0.1183* (−3.3%)	0.1203 (−1.7%)

Table 3 Average precision of different methods with overlap rate between 40 and 60%([0,1] linear normalization and Round-robin serves as baseline)

Docs rank	Round-robin	MNZ	Bayesian ($k = 2.5$)	Borda ($k = 0.95$)	SDM ($k = 0.5$)	MEM
5	0.2782	0.2812 (+1.6%)	0.2950* (+6.1%)	0.2794 (+0.6%)	0.3157* (+13.5%)	*.3163** (+13.7%)
10	0.2612	0.2411* (−7.7%)	0.2618 (+0.2%)	0.2495* (−4.5%)	0.2746* (+5.1%)	*0.2772** (+6.2%)
15	0.2376	0.2179* (−8.3%)	0.2385 (−0.4%)	0.2279* (−4.1%)	0.2505* (+5.4%)	*0.2541** (+6.9%)
20	0.2250	0.1995* (−11.3%)	0.2189 (−2.7%)	0.2085* (−7.3%)	0.2328* (+3.3%)	*0.2335** (+3.8%)
25	0.2130	0.1860* (−12.7%)	0.2039* (−4.3%)	0.1940* (−8.9%)	0.2161 (+1.5%)	*0.2182* (+2.4%)
30	0.1967	0.1750* (−11.0%)	0.1901 (−3.4%)	0.1836* (−6.7%)	0.2047* (+4.1%)	*0.2063** (+4.9%)
50	0.1624	0.1489* (−8.3%)	0.1604 (−1.2%)	0.1546* (−4.8%)	0.1726* (+6.3%)	*0.1739** (+7.1%)
100	0.1264	0.1143* (−9.6%)	0.1213* (−4.0%)	0.1153* (−8.8%)	0.1277 (+1.0%)	*0.1293* (+2.3%)

Table 4 Average precision of different methods with overlap rate between 60 and 80% ([0,1] linear normalization and Round-robin serves as baseline)

Docs rank	Round-robin	MNZ	Bayesian ($k = 2.5$)	Borda ($k = 0.95$)	SDM ($k = 0.5$)	MEM
5	0.2760	0.2939* (+6.5%)	0.3013* (+9.2%)	0.2819 (+2.1%)	*0.3139** (+13.7%)	0.3083* (+11.7%)
10	0.2571	0.2516 (−2.1%)	0.2569 (± 0.0%)	0.2445* (−4.9%)	0.2740* (+6.6%)	*0.2759** (+7.3%)
15	0.2363	0.2212* (−6.4%)	0.2365 (±0.0%)	0.2223* (−5.9%)	0.2488* (+5.3%)	*0.2500** (+5.8%)
20	0.2223	0.2009* (−9.6%)	0.2163 (−2.7%)	0.2086* (−6.1%)	0.2316* (+4.2%)	*0.2334** (+4.9%)
25	0.2081	0.1877* (−9.8%)	0.2048* (−1.6%)	0.1936* (−7.0%)	*0.2172** (+4.4%)	*0.2172** (+4.4%)
30	0.1948	0.1778* (−8.7%)	0.1880* (−3.5%)	0.1823* (−6.4%)	0.2035* (+4.7%)	*0.2059** (+5.6%)
50	0.1610	0.1494* (−7.2%)	0.1597 (−0.8%)	0.1548* (−3.9%)	0.1720* (+6.8%)	*0.1736** (+7.8%)
100	0.1259	0.1115* (−11.4%)	0.1180* (−6.3%)	0.1149* (−8.7%)	0.1275 (+1.3%)	*0.1290** (+2.5%)

Table 5 Average precision of different methods with overlap rate between 80 and 100% ([0,1] linear normalization and Round-robin serves as baseline)

Docs rank	Round-robin	MNZ	Bayesian ($k = 2.5$)	Borda ($k = 0.95$)	SDM ($k = 0.5$)	MEM
5	0.2738	0.3032* (+10.7%)	0.3156* (+15.3%)	0.2956* (+8.0%)	0.3120* (+14.0%)	*0.3188** (+16.4%)
10	0.2600	0.2691 (+3.5%)	0.2678 (+3.0%)	0.2482* (−4.5%)	*0.2753** (+5.9%)	0.2726* (+4.8%)
15	0.2390	0.2399 (+0.4%)	0.2411 (+0.9%)	0.2224* (−6.9%)	*0.2458** (+2.8%)	0.2443 (+2.2%)
20	0.2258	0.2174 (−3.7%)	0.2234 (−1.1%)	0.2064* (−8.6%)	0.2292 (+1.5%)	*0.2296* (+1.7%)
25	0.2129	0.2032* (−4.6%)	0.2082 (−2.2%)	0.1908* (−10.4%)	0.2149 (+0.9%)	*0.2163* (+1.6%)
30	0.1991	0.1893* (−4.9%)	0.1946 (−2.3%)	0.1805* (−9.3%)	*0.2044* (+2.7%)	0.2039 (+2.4%)
50	0.1635	0.1612 (−1.4%)	0.1656 (+1.3%)	0.1521* (−7.0%)	*0.1731** (+5.9%)	0.1724* (+5.4%)
100	0.1263	0.1214 (−3.9%)	0.1238 (−2.0%)	0.1124* (−11.0%)	*0.1300* (+2.9%)	0.1298 (+2.8%)

Table 6 Average precision of different methods with overlap rate between 0 and 20% (regression, Round-robin serves as baseline)

Docs rank	Round-robin	MNZ	SDM ($k = 0.5$)	MEM
5	*0.2676*	0.2200* (−17.8%)	0.2488* (−7.0%)	0.2556* (−4.5%)
10	*0.2437*	0.1830* (−24.9%)	0.2194* (−10.0%)	0.2126* (−12.8%)
15	*0.2228*	0.1559* (−30.0%)	0.1922* (−13.7%)	0.1881* (−15.6%)
20	*0.2079*	0.1400* (−32.7%)	0.1730* (−16.8%)	0.1685* (−19.0%)
25	*0.1953*	0.1282* (−34.4%)	0.1598* (−13.7%)	0.1570* (−15.2%)
30	*0.1830*	0.1184* (−35.3%)	0.1491* (−18.5%)	0.1476* (−19.3%)
50	*0.1542*	0.0948* (−38.5%)	0.1208* (−21.7%)	0.1178* (−23.6%)
100	*0.1181*	0.0680* (−42.4%)	0.0882* (−25.3%)	0.0855* (−27.6%)

Table 7 Average precision of different methods with overlap rate between 20 and 40% (regression, Round-robin serves as baseline)

Docs rank	Round-robin	MNZ	SDM ($k = 0.5$)	MEM
5	*0.2844*	0.2150*	0.2616*	0.2658*
		(−24.4%)	(−8.0%)	(−6.5%)
10	*0.2584*	0.1709*	0.2245*	0.2277*
		(−33.9%)	(−13.1%)	(−11.9%)
15	*0.2352*	0.1493*	0.1945*	0.1987*
		(−36.5%)	(−17.3%)	(−15.5%)
20	*0.2169*	0.1336*	0.1769*	0.1786*
		(−38.9%)	(−23.1%)	(−22.2%)
25	*0.2015*	0.1227*	0.1521*	0.1548*
		(−39.1%)	(−24.5%)	(−23.2%)
30	*0.1899*	0.1148*	0.1421*	0.1433*
		(−39.5%)	(−25.2%)	(−24.5%)
50	*0.1596*	0.0939*	0.1223*	0.1199*
		(−41.2%)	(−23.4%)	(−24.9%)
100	*0.1224*	0.0753*	0.0919*	0.0931*
		(−38.5%)	(−24.9%)	(−23.9%)

Table 8 Average precision of different methods with overlap rate between 40 and 60% (regression, Round-robin serves as baseline)

Docs rank	Round-robin	MNZ	SDM ($k = 0.5$)	MEM
5	0.2782	0.2729	*0.3071**	0.2905*
		(−1.9%)	(+10.4%)	(+4.4%)
10	0.2612	0.2350*	*0.2690**	0.2555
		(−10.0%)	(+3.0%)	(−2.2%)
15	0.2376	0.2113*	*0.2397*	0.2265*
		(−11.6%)	(+0.9%)	(−4.7%)
20	*0.2250*	0.1902*	0.2167*	0.2116*
		(−15.5%)	(−3.7%)	(−6.0%)
25	*0.2130*	0.1772*	0.1990*	0.1953*
		(−16.8%)	(−6.6%)	(−8.3%)
30	*0.1967*	0.1635*	0.1822*	0.1770*
		(−16.9%)	(−7.4%)	(−10.0%)
50	*0.1624*	0.1340*	0.1502*	0.1512*
		(−17.5%)	(−7.5%)	(−6.9%)
100	*0.1264*	0.0949*	0.1105*	0.1090*
		(−24.9%)	(−12.6%)	(−13.8%)

Table 9 Average precision of different methods with overlap rate between 60 and 80% (regression, Round-robin serves as baseline)

Docs	Round-robin	MNZ	SDM ($k = 0.5$)	MEM
5	0.2760	0.2947*	*0.3064**	0.2960*
		(+6.8%)	(+11.0%)	(+7.2%)
10	0.2571	0.2469*	*0.2754**	0.2618
		(−4.0%)	(+7.1%)	(+1.8%)
15	0.2363	0.2181*	*0.2406*	0.2322
		(−7.7%)	(+1.8%)	(−1.7%)
20	*0.2223*	0.1957*	0.2183	0.2131
		(−12.0%)	(−1.8%)	(−4.1%)
25	*0.2081*	0.1792*	0.2018*	0.1969*
		(−13.0%)	(−3.0%)	(−5.4%)
30	*0.1948*	0.1675*	0.1905	0.1835*
		(−14.0%)	(−2.2%)	(−5.8%)
50	*0.1610*	0.1363*	0.1585	0.1496*
		(−15.3%)	(−1.5%)	(−7.1%)
100	*0.1259*	0.0966*	0.1133*	0.1104*
		(−23.2%)	(−10.0%)	(−12.3%)

Table 10 Average precision of different methods with overlap rate between 80 and 100% (regression, Round-robin serves as baseline)

Docs rank	Round-robin	MNZ	SDM ($k = 0.5$)	MEM
5	0.2738	0.3178*	*0.3248**	0.3132*
		(+16.1%)	(+18.6%)	(+14.4%)
10	0.2600	0.2824*	*0.2934**	0.2760*
		(+8.6%)	(+12.8%)	(+6.2%)
15	0.2390	0.2511*	*0.2588**	0.2512*
		(+5.1%)	(+8.3%)	(+5.1%)
20	0.2258	0.2264	*0.2363**	0.2283
		(+0.3%)	(+4.7%)	(+1.1%)
25	0.2129	0.2072	*0.2175*	0.2109
		(−2.7%)	(+2.2%)	(−0.9%)
30	0.1991	0.1921*	*0.2037*	0.1961
		(−3.5%)	(+2.3%)	(−1.5%)
50	0.1635	0.1598	*0.1700**	0.1656
		(−2.3%)	(+4.0%)	(+1.3%)
100	*0.1263*	0.1155*	0.1235*	0.1229*
		(−8.6%)	(−2.2%)	(−2.7%)

Acknowledgments The authors thank the comments and helpful suggestions of the anonymous referees.

References

Aslam, J. A., & Montague, M. (2003). Models for metasearch. In: *Proceedings of the 24th Annual International ACM SIGIR Conference* (pp. 24–37). ACM, Gaithersburg, MD.

Baeza-Yates, R., & Ribeiro-Neto, B. (1999). *Modern information retrieval*. Reading, MA: ACM and Addison-Wesley.

Callan, J. (2000). Distributed information retrieval. *Advances in Information Retrieval* (pp. 127–150), Kluwer Academic Publishers.

Callan, J. K., Lu, Z., & Croft, W. (1995). Searching distributed collections with inference networks. In *Proceedings of the 18th Annual International ACM SIGIR Conference* (pp. 21–28). ACM, Seattle, WA.

Calvé, A. L., & Savoy, J. (2000). Database merging strategy based on logistic regression. *Information Processing and Management*, *36*(3), 341–359.

Chowdhury, A., Frider, O., Grossman, D., & McCabe, C. (2002). Collection statistics for fast duplicate document detection. *ACM Transactions on Information Systems*, *20*(2), 171–191.

Fox, E. A., & Shaw, J. (1994). Combination of multiple searches. In *The Second Text REtrieval Conference (TREC-2)* (pp. 243–252). National Institute of Standards and Technology, USA, Gaitherburg, MD.

Fuhr, N. (1999). A decision-theoretic approach to database selection in networked IR. *ACM Transactions on Information Systems*, *17*(3), 229–249.

Gauch, S., Wang, G., & Gomez, M. (1996). Profusion: Intelligent fusion from multiple, distributed search engines. *Journal of Universal Computer Science*, *2*(9), 637–649.

Gravano, L., García-Molina, H., & Tomasic, A. (1999). Gloss: Text-source discovery over the Internet. *ACM Transactions on Database Systems*, *24*(2), 229–264.

Hawking, D., & Robertson, S. (2003). On collection size and retrieval effectiveness. *Information Retrieval*, *9*(1), 99–105.

Hawking, D., & Thistlewaite, P. (1999). Methods for information server selection. *ACM Transactions on Information Systems*, *17*(1), 40–76.

Hull, D. A., Pedersen, J. O., & Schüze, H. (1996). Method combination for document filtering. In *Proceedings of the 19th Annual International ACM SIGIR Conference* (pp. 279–287). ACM, Zurich, Switzerland.

Lawrence, S., & Giles, C. L. (1998a). Context and page analysis for improved Web search. *IEEE Internet Computing*, *2*(4), 38–46.

Lawrence, S., & Giles, C. L. (1998b). Inquiris, the NECI search engine. *Computer Networks (Proceedings of WWW 7)*, *30*(1–7), 95–105.

Lawrence, S., & Giles, C. L. (2000). Accessibility of information on the Web. *Intelligence*, *11*(1), 32–39.

Lee, J. H. (1997). Analysis of multiple evidence combination. In *Proceedings of the 20th Annual International ACM SIGIR Conference* (pp. 267–275). ACM, Philadelphia, PA.

Lemur, retrieved on October 25, 2005 from http://www.lemurproject.org/

Lucene, retrieved on October 25, 2005 from http://lucene.apache.org/java/docs/index.html

Manmatha, R., Rath, T., & Feng, F. (2001). Modeling score distributions for combining the outputs of search engines. In *Proceedings of the 24th Annual International ACM SIGIR Conference* (pp. 267–275). ACM, New Orleans, LA.

Meng, W., Yu, C., & Liu, K. (2002). Building efficient and effective metasearch engines. *ACM Computing Surveys*, *34*(1), 48–89.

Montague, M. (2002). *Metasearch: Data fusion for document retrieval* (Technical Report TRC2002-424). Hanover, NH: Department of Computer Science, Dartmouth College.

Montague, M., & Aslam, J. A. (2002). Condorcet fusion for improved retrieval. In *Proceedings of ACM CIKM Conference* (pp. 538–548). ACM, McLean, VA.

Oztekin, B., Karypis, G., & Kumar, V. (2002). Expert agreement and content based reranking in a meta search environment using Mearf. In *Proceedings of the 11th International World Wide Web Conference* (pp. 333–344). ACM, Honolulu, HI.

Rasolofo, Y., Hawking, D., & Savoy, J. (2003). Result merging strategies for a current news metasearcher. *Information Processing and Management*, *39*(4), 581–609.

Searchengineshowdown, retrieved on October 25, 2005 from http: //www.searchengineshowdown.com/stats/overlap.shtml

Selberg, E., & Etzioni, O. (1995). Multi-service search and comparison using the MetaCrawler. In *Proceedings of the 4th International World Wide Web Conference*. Darmstadt, Germany.

Selberg, E., & Etzioni, O. (1997). The MetaCrawler architecture for resource aggregation on the Web. *IEEE Expert*, *12*(1), 11–14.

Selberg, E., & Etzioni, O. (2000). On the instability of web search engines. In *Proceedings of RIAO conference* (pp. 223–235). Paris, France.

Si, L., & Callan, J. (2002). Using sampled data and regression to merge search engine results. In *Proceedings of the 25th Annual International ACM SIGIR Conference* (pp. 19–26). ACM, Tampere, Finland.

Vogt, C. C., & Cottrell, G. A. (1999). Fusion via a linear combination of scores. *Information Retrieval*, *1*(3), 151–173.

Voorhees, E.M., Gupta, N.K., & Johnson-Laird, B. (1995). Learning collection fusion strategies. In *Proceedings of the 18th Annual International ACM SIGIR Conference* (pp. 172–179). ACM, Seattle, WA.

Voorhees, E. M., & Harman, D. K. (Eds.). (1996). In *Proceedings of the 5th Text Retrieval Conference*. National Institute of Standards and Technology, USA, Gaithersburg, MD.

Wu, S., & Crestani, F. (2003). Distributed information retrieval: A multi-objective resource selection approach. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, *11*(Suppl.), 83–100.

Wu, S., & Crestani, F. (2004). Shadow document methods of results merging. In *Proceedings of the 19th ACM Symposium on Applied Computing* (pp. 1067–1072). ACM, Nicosia, Cyprus.

Yuwono, B., & Lee, D. (1997). Server ranking for distributed test retrieval systems on the Internet. In *Proceedings of the Fifth International Conference on Database Systems for Advanced Application* (pp. 41–50). World Scientific, Melbourne, Australia.