Evaluating Server Selection for Federated Search

Paul Thomas[1] and Milad Shokouhi[2]

[1] CSIRO paul.thomas@csiro.au
[2] Microsoft Research milads@microsoft.com

Abstract. Previous evaluations of server selection methods for federated search have either used metrics which are unconnected with user satisfaction, or have not been able to account for confounding factors due to other search components.
We propose a new framework for evaluating federated search server selection techniques. In our model, we isolate the effect of other confounding factors such as server summaries and result merging. Our results suggest that state-of-the-art server selection techniques are generally effective but result merging methods can be significantly improved. Furthermore, we show that the performance differences among server selection techniques can be obscured by ineffective merging.

1 Introduction

Federated search attempts to provide a single interface to several independent search engines. A *broker* translates a user's query, forwards it to one or more search engines, and presents any results in a single display [2].

Brokers commonly include a process of *server selection* to determine which search engines should receive each query. By using the most appropriate set of search engines, a broker can reduce direct costs, network traffic, and processing requirements; there is also some evidence that it can increase effectiveness even over that of a single, non-federated, system [1]. A large number of server selection methods have been proposed in the literature, and with thorough evaluations it should be possible to answer two questions: first, what methods should be considered for a new broker? Second, is the selection method in an existing broker performing well?

Selection evaluation unfortunately remains problematic. There are at least three desiderata for evaluation: (1) the measure should somehow reflect user experience; (2) it should be possible to identify how well server selection is doing, without confounding factors of other broker tasks; and (3) it should be possible to measure effectiveness across a range of external factors.

The first of these criteria has been satisfied by using a metric such as precision at ten documents (P@10), measured on a broker's final list of results [e.g. 4]. Although this does reflect user-visible quality, it does not allow us to find out how effective the broker's selection (or any other) component is—as well as selection, P@10 results are influenced by query translation, merging results, and other steps. Since the effect of each component cannot be separated it is also impossible to understand which parts of a broker should receive attention.

Other evaluations have satisfied the second criterion by using a metric which considers only the effectiveness of selection—commonly $\mathcal{R}_k$, which is an analogue of recall measured just after selection but before any further processing by search engines or the broker [3]. However, there is no evidence to suggest that $\mathcal{R}_k$ correlates with any user-visible effectiveness and some authors have seen $\mathcal{R}_k$ decline as P@10 is stable [8]. Further, since $\mathcal{R}_k$ does not depend on the quality of retrieval at each search engine, it is not possible to understand how selection effectiveness relates to search engine effectiveness—violating our third criterion.

We propose a novel method which isolates the effects of broker and search engine components, and which satisfies the criteria above.

2 Evaluating Server Selection

To satisfy the first of our two desirable characteristics, we propose evaluating systems with respect to P@10.

To reconcile the remaining characteristics, we propose two steps. First, we attempt to remove the confounding effect of other broker tasks. This isolates any performance impact due to the selection process itself; with confounding effects removed, any change in performance is due to selection alone. Second, we consider a range of possible cases for those aspects outside the broker's control.

There are several subsystems in a broker: creating summaries of a server's holdings (probably by taking samples of documents); selection; query translation; parsing result sets; and merging results into a final list. Outside the broker's control, but having considerable impact, is the retrieval at each selected engine.

Removing the Effect of Server Summaries. Errors or inaccuracies in server summaries will inevitably impact server selection [7]. To remove the effect of server summaries, we assume that selection should only consider those servers where an accurate summary would match the query; for the methods in our experiments, this is equivalent to using full term statistics for the "summaries".

Removing the Effect of Query Translation and Result Set Parsing. We simply assume these steps are possible without any loss of fidelity. (This is a common assumption in the federated search literature, although not often acknowledged.)

Removing the Effect of Merging. After servers are selected, and have retrieved results, there is a further possible confounding effect from the process of merging the returned results. Most merging algorithms are "stable"—that is, if there is no overlap between returned result lists then they will respect each server's ordering. Since servers may be ineffective and return irrelevant documents before relevant ones, we control for the confounding effect by assuming "perfect" merging, which instead puts all relevant documents at the head of the merged list.

Best Possible Performance. We evaluate selection using each method's *best possible performance* (BPP). BPP at a given point in a broker's working is the performance measure it could achieve if every subsequent component works without error, given a particular configuration of those things outside the broker's control (Figure 1(a)). Observe that initially, BPP is fixed at the best possible performance. After building summaries, BPP is still maximum (since, at least in theory, the rest of the broker can still get the right documents), but any later drop reflects the performance of successive steps. The drop in BPP at each step isolates the effectiveness of that component (our second desideratum); so the drop at point (a) is the loss in effectiveness due to server selection, which a broker can control, and that at point (b) is that due to retrieval, which it cannot.

Here, we are interested in selection alone so we remove the confounding effect of other components, as above; in effect, they make no difference to BPP. We use P@10 as the performance measure and vary the query, the testbed, and the effectiveness of each search engine (this satisfies our second desideratum).

3 Experiments

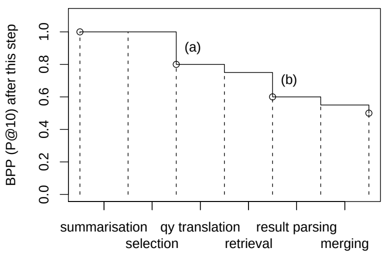
In these initial experiments, we use the "uniform" testbed of Xu and Croft, which features 100 "collections" formed from TREC ad-hoc data [9], and TREC topics 51–100. Each server is assumed to return up to ten documents, and up to ten servers are selected for each query. We consider the CRCS(e) [5] and ReDDE [6] selection algorithms, as well as SUSHI [8] selecting for precision ("SUSHI") and recall ("SUSHI.R").

What is the Impact of Selection? Our first experiments consider the BPP of a broker using each of the three algorithms, assuming perfect retrieval—all relevant documents are returned before any non-relevant—as well as perfect performance of other broker components. Since all topics have at least ten relevant documents, any drop in P@10 from 1.0 can be attributed to inaccuracy in server selection.

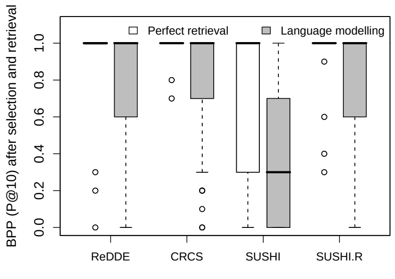
The left-hand boxes in Figure 1(b) illustrate best possible P@10 in this scenario. ReDDE, CRCS, and SUSHI.R are all capable of near-perfect selection; SUSHI, which selects many fewer servers, performs much more poorly.

Previous experiments have shown no difference between these methods on P@10, but have not accounted for errors introduced by other processing. The difference between the BPP reported here and the P@10 figures reported elsewhere [5; 6; 8]—around 0.4 ± 0.05—is due to ineffective merging. As well as obscuring the high performance of these selection algorithms, relatively poor merging has also obscured the differences between SUSHI and other methods.

How Robust are the Techniques? The right-hand side (grey) boxes in Figure 1(b) illustrate BPP after selection and retrieval when selected servers use a standard language modelling retrieval engine instead of perfect retrieval. Comparing the two sets of data allows a comparison of broker effectiveness, for each method, as one aspect of the broker's environment changes.



(a) BPP (P@10) after each step



(b) BPP (P@10) after retrieval

Fig. 1. Using Best Possible Performance (BPP) to investigate search effectiveness

Here it is clear that retrieval effectiveness does impact BPP for each of the four algorithms, but (again with the exception of SUSHI) P@10 scores are still fairly high. This is expected—ad-hoc retrieval should not be so poor as to make a large impact. The BPP here is still well above P@10 figures for whole systems, however, so it still seems that poor merging is responsible for much of the poor effectiveness observed in earlier work.

4 Conclusions

By considering a broker's best possible performance, it is possible to evaluate server selection with a user-visible metric; in isolation from other components; and with regard to a number of factors outside the broker's control. Early experiments suggest that modern selection algorithms perform very well, and that the performance seen in complete brokers is likely due to poor merging in particular.

References

[1] Faïza Abbaci, Jacques Savoy, and Michel Beigbeder. A methodology for collection selection in heterogeneous contexts. In *Proc. IEEE Conference on Information Technology*, 2002.
[2] Jamie Callan. Distributed information retrieval. In W Bruce Croft, editor, *Advances in information retrieval*, volume 7 of *The information retrieval series.* Springer, 2000.
[3] Luis Gravano and Héctor García-Molina. Generalizing GlOSS to vector-space databases and broker hierarchies. In *Proc. VLDB*, 1995.
[4] Allison L. Powell, James C. French, Jamie Callan, Margaret Connell, and Charles L. Viles. The impact of database selection on distributed searching. In *Proc. ACM SIGIR*, 2000.
[5] Milad Shokouhi. Central-rank-based collection selection in uncooperative distributed information retrieval. In *Proc. ECIR*, 2007.
[6] Luo Si and Jamie Callan. Relevant document distribution estimation method for resource selection. In *Proc. ACM SIGIR*, 2003.
[7] Paul Thomas and David Hawking. Server selection methods in personal metasearch: A comparative empirical study. *Information Retrieval*, 12:581–604, 2009.
[8] Paul Thomas and Milad Shokouhi. SUSHI: Scoring scaled samples for server selection. In *Proc. ACM SIGIR*, 2009.
[9] Jinxi Xu and W. Bruce Croft. Cluster-based language models for distributed retrieval. In *Proc. ACM SIGIR*, 1999.