# Harvesting All Matching Information To A Given Query From a Deep Website

Mohammadreza Khelghati[1], Djoerd Hiemstra[1], and Maurice van Keulen[1]
{s.m.khelghati, d.hiemstra, m.vankeulen}@utwente.nl

[1]Databases Group, University of Twente
Enschede, Netherlands

**Abstract.** In this paper, the goal is harvesting all documents matching a given (entity) query from a deep web source. The objective is to retrieve all information about for instance "Denzel Washington", "Iran Nuclear Deal", or "FC Barcelona" from data hidden behind web forms. Policies of web search engines usually do not allow accessing all of the matching query search results for a given query. They limit the number of returned documents and the number of user requests. In this work, we propose a new approach which automatically collects information related to a given query from a search engine, given the search engine's limitations. The approach minimizes the number of queries that need to be sent by applying information from a large external corpus. The new approach outperforms existing approaches when tested on Google, measuring the total number of unique documents found per query.

## 1 Introduction

The goal of this research is to harvest all documents matching a given (entity) query from a deep web source. For instance, we aim at retrieving all information about "Denzil Washington", "Iran Nuclear Deal", or "FC Barcelona" from data hidden behind web forms. However, policies of search engines usually do not allow accessing all of the matching query search results for a given query. They limit the number of returned documents (#*ResultsLimited*) and the number of user requests (#*RequestsLimited*).

Given these search engines limitations, we propose a new approach which automatically collects information related to a given query from a search engine. To do so, we rely on search refinement techniques to uncover results beyond what a search engine allows a user to directly access due to #`ResultsLimited` and #`RequestsLimited` limitations. These techniques are typically based on adding extra terms to the initial query to obtain refined search results. We propose an approach which refines search results for the purpose of achieving full data coverage.

In this approach, reformulating queries should be carried out with the aim of obtaining as many new results as possible for each query. Maximizing the number of new results means submitting queries which return as many documents as #`ResultsLimited` limitation allows while minimizing the number of duplicates.

Minimizing duplicates becomes complicated with the presence of *ranking bias* and *query bias* [3]. Search engine's ranking algorithms (e.g. Google Page-rank) and selection of the initial query favour some documents more than others to be returned by the search engine.

To meet this challenge, techniques from *Deep Web harvesting* [1, 12, 10, 4, 2], *Query-Based Sampling* [5, 3], *Topical Crawling*, [14, 16], and *Query Expansion* [6, 13, 7–9] are studied. Based on these studies, several approaches are suggested, implemented, and compared in this paper. We test our approaches on Google, which claims to search 100 PB of Web data (60 trillion URLs)[1]. Google imposes both #`ResultsLimited` and #`RequestsLimited`, and ranking bias through its Page-Rank algorithm.

## 2 Suggested Approach

To reach this data coverage, we send automatically generated queries to a search engine's API with the goal of retrieving all documents that contain a given entity with a minimum amount of query submissions. We compare the approaches by their capabilities to deal with #`ResultsLimited` and #`RequestsLimited`. The comparison is based on the average number of queries submitted to retrieve all documents for a given query.

We distinguish two kinds of approaches. Section 2.1 describes ideal approaches, for which we estimate the number of queries needed in ideal (simulated) conditions. Section 2.2 describes approaches in which queries are reformulated by using an external corpus.

### 2.1 Ideal Approaches

The approach mentioned in this section is desirable or perfect but not easily realized. This is investigated with the sole purpose of improving the comparison of the introduced approaches.

**Oracle Perfect Approach** To achieve a full data coverage on a given entity in a search system with the #`ResultsLimited` and #`RequestsLimited` limitations, the perfect approach is the one which returns not only the maximum possible number of documents but also only unique ones for each request. To have a complete coverage in this situation, it is adequate to send only the $\frac{|CollectionSizeForQuery|}{allowedDocsToBeVisited}$ number of requests. In reality, this is not easily reachable. To do so, you need to know the exact mechanism of search engine ranking algorithm. Then, you might be able to divide the collection into exactly $\frac{||CollectionSizeForQuery||}{allowedDocsToBeVisited}$ sub-collections. In addition to the knowledge of ranking algorithm, you might need additional information. For instance, if a ranking algorithm is based on terms frequencies, you need to know all the term frequencies

[1] Official Google Blog: http://googleblog.blogspot.nl/2008/07/we-knew-web-was-big.html

beforehand. This kind of information is only accessible when you have full index access.

## 2.2 List-Based Query Generation Approach

In these approaches, the terms to be added to the seed query are selected from a list of words. This list is generated from an external corpus and includes the frequencies in that corpus. In this paper, this list is extracted from the ClueWeb09 dataset, which is a web crawl containing nearly 500 million English pages [15]. Selecting terms from the list of terms and their corresponding document frequencies can be performed in different methods. In the following, these methods are further explained and studied.

**List-Based Most/Least Frequent Approach** Although primitive, choosing the most or least frequent words from a list are possible options in selecting terms. As the ClueWeb dataset is not a topic-specific corpus, the most frequent words from this corpus are highly probable to be also general in all other not topic-specific corpora.

**Pre-determined Frequency Based Approach** While submitting the most frequent terms increases the chance of reaching the maximum number of returned results and the least frequent ones increases the probabilities of generating fewer duplicates, it is of a great interest to investigate the likelihood of finding a term frequency which creates a trade-off between these two. To do so, statistical formulas are applicable. If events $A$ and $B$ are independent, then the probability of them both occurring is the product of the probabilities of each occurring ($P(A\&B) = P(A) * P(B)$). With samples smaller than 10 percent of the collection, we can assume two posing query processes as statistically independent events ("The 10% Condition"). Then, the probability of having an overlap between two queries equals with the multiplication of the probability of each query. This is shown in Formula 1.

$$\frac{|MatchingDocs \cap ReturnedDocs|}{|SearchEngine|} = \frac{|MatchingDocs|}{|SearchEngine|} * \frac{|ReturnedDocs|}{|SearchEngine|}$$
$$|ReturnedDocs| = \frac{l * |SearchEngine|}{|MatchingDocs|} \mid (|MatchingDocs \cap ReturnedDocs| = l) \tag{1}$$

With the knowledge of targeted search engine's index size, and also the number of documents matching seed query, through Formula 1, one can determine the frequency of another query for which the overlap of this query and the seed query equals the number of documents allowed to be visited. This means with information on the seed query, returned results and search engine size, a term can be found to formulate a new query returning at least the same number of results that are allowed to be visited. This enables avoiding the permanent presence of the same highly ranked documents among the results and creates a higher chance in collecting more new documents in each trial. If the size of search engine is unknown, as discussed in [11], the size can be estimated by only using a few number of generated samples from search engine.

As pointed out, applying this formula to our case requires information on terms document frequencies. To access this information from the targeted search system, we should download all its content and count all the terms document frequencies. If this was possible, there was no need for introducing these approaches. Instead, we can use pre-computed terms document frequencies from an external corpus. In this paper, as we test our approaches on Google, we use the ClueWeb dataset. However, the size difference between the ClueWeb and Google should be considered to be able to apply the formula. The easiest solution is to include different sizes in the calculations. For example, assuming $Size^{SearchEngine} = 10^9$, the number of English documents in ClueWeb as $5 \times 10^8$, $limitedResults = 100$, and $|MatchingDocuments|$ for a given query to be $4 \times 10^5$, the following calculation could provide us with a term document frequency that has higher chance to result in samples of our desired size: $\frac{100}{10^9} = \frac{4\times10^5}{10^9} * \frac{x}{5\times10^8} \Longrightarrow x = 125000$. In this paper, we refer to this approach as *LB-FixedFreq.* approach.

## 3 Experiments and Results

### 3.1 Experiments Settings

*Test Search Engine* In these experiments, Google as the biggest web search engine with one of the most complicated ranking algorithms is considered as our test search engine. As the only necessary feature for applying any of these suggested approaches is the support of keyword-search interface, targeting Google does not limit our findings only to Google.

*Entities Test Set* In our experiments, we used four different entities ("Vitol", "Ed Brinksma", "PhD Comics", and "Fireworks Disaster") to test and compare the suggested approaches. We tried to include entities representing different types of entities; Company, Person, Topic, and Event. In addition to difference in type, we tried to cover queries with different estimated results sets sizes.

### 3.2 Results

In this section, the results of applying the introduced approaches in Section 2 to the test entities (Section 3.1) are presented. The Figure 1 compares the performances of all the approaches for one of the test entities in the test set. This is a straight forward task as it is only required to compare the number of retrieved documents by each approach. However, to compare the approaches' performances on all the test cases, we calculate their average distances from the Oracle Perfect approach. In Figure 3, the performances of all the approaches for all the entities in the test set are compared with the Oracle Perfect approach.

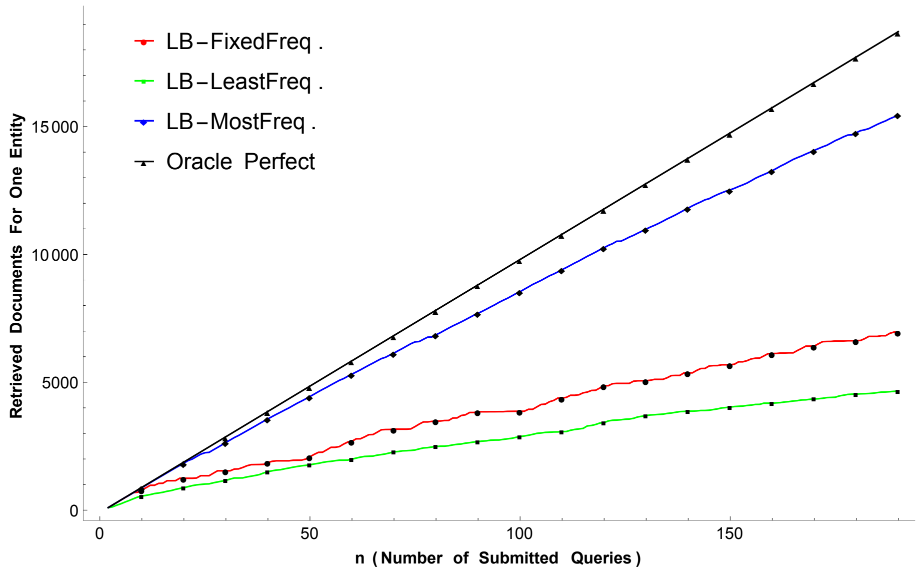

**Fig. 1.** Average Performance For All One Entity

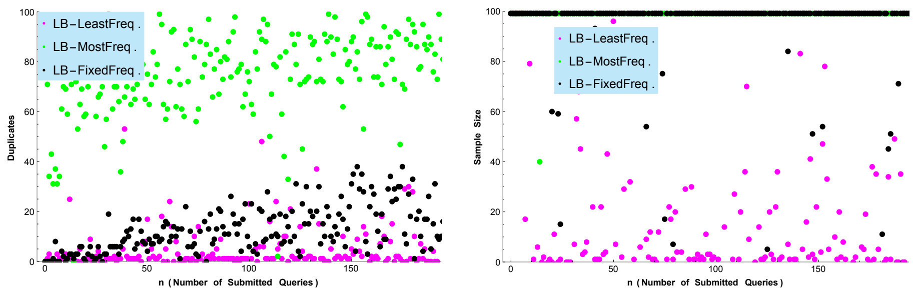

**Fig. 2.** Sample Sizes and Duplicates For Approaches For One Entity

As it is shown in Fig 3, the LB-FixedFreq. approach performs better than Most-freq. and Least-Freq. approaches. This approach submits queries which result in fewer duplicates than LB-MostFreq. approach while having bigger sample sizes in regards to the Least-Freq approach. This is observable from Figure 2. The right image in this figure shows the number of duplicates resulted from submitting all the queries formulated by adding a term to the initial query (given entity). The left picture shows the corresponding sample size for each of these queries. From comparing these two images, we can conclude that a trade-off between the big sample sizes and number of duplicates is the key to the LB-MostFreq. approach's better performance. In this approach, finding a specific frequency leads to a trade-off between sample sizes and number of duplicates.

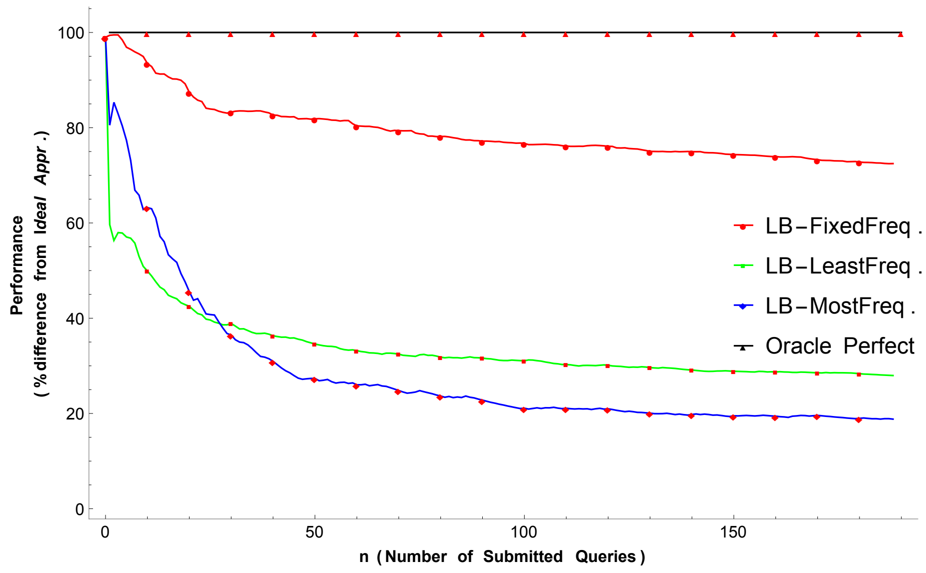

**Fig. 3.** Average Performance For All The Entities

## 4 Conclusion and Future Work

In this work, we assessed different query generation mechanisms for harvesting a web data source to move forward towards achieving a full data coverage on a given (entity) query. From the experiments, we found that the key to success in these approaches is to send queries which result in the maximum possible number of results with the minimum possible number of documents downloaded in previous query submissions. To have this success factor, we suggested three different approaches based on different frequencies. Among these approaches, the LB-FixedFreq. performed better than the others.

*Future Work* In addition to the frequency of terms extracted from an external corpus, we can include terms present in the previously retrieved documents to select the best next query to submit. The frequency of these terms could also be applied for a more efficient query expansion technique.

## 5 Acknowledgement

This publication was supported by the Dutch national program COMMIT.

## References

1. Manuel Álvarez, Juan Raposo, Alberto Pan, Fidel Cacheda, Fernando Bellas, and Víctor Carneiro. Deepbot: a focused crawler for accessing hidden web content.

In *Proceedings of the 3rd international workshop on Data engineering issues in E-commerce and services: In conjunction with ACM Conference on Electronic Commerce (EC '07)*, DEECS '07, pages 18–25, New York, NY, USA, 2007. ACM.

2. Luciano Barbosa and Juliana Freire. Siphoning hidden-web data through keyword-based interfaces. In *SBBD*, pages 309–321, 2004.
3. Krishna Bharat and Andrei Broder. A technique for measuring the relative size and overlap of public web search engines. *Comput. Netw. ISDN Syst.*, 30:379–388, April 1998.
4. Michael Cafarella. Extracting and Querying a Comprehensive Web Database. In *Proceedings of the Conference on Innovative Data Systems Research (CIDR)*, 2009.
5. James P. Callan and Margaret E. Connell. Query-based sampling of text databases. *ACM Trans. Inf. Syst.*, 19(2):97–130, 2001.
6. Guihong Cao, Jian-Yun Nie, Jianfeng Gao, and Stephen Robertson. Selecting good expansion terms for pseudo-relevance feedback. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, pages 243–250, New York, NY, USA, 2008. ACM.
7. Claudio Carpineto and Giovanni Romano. A survey of automatic query expansion in information retrieval. *ACM Comput. Surv.*, 44(1):1:1–1:50, January 2012.
8. Kevyn Collins-Thompson and Jamie Callan. Estimation and use of uncertainty in pseudo-relevance feedback. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '07, pages 303–310, New York, NY, USA, 2007. ACM.
9. Ben He and Iadh Ounis. Combining fields for query expansion and adaptive query expansion. *Inf. Process. Manage.*, 43(5):1294–1307, September 2007.
10. Yeye He, Dong Xin, Venkatesh Ganti, Sriram Rajaraman, and Nirav Shah. Crawling deep web entity pages. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, WSDM '13, pages 355–364, New York, NY, USA, 2013. ACM.
11. Mohammadreza Khelghati, Djoerd Hiemstra, and Maurice van Keulen. Size estimation of non-cooperative data collections. IIWAS '12, pages 239–246, New York, NY, USA, 2012. ACM.
12. Jayant Madhavan, David Ko, Lucja Kot, Vignesh Ganapathy, Alex Rasmussen, and Alon Halevy. Google's Deep Web crawl. *Proc. VLDB Endow.*, 1(2):1241–1252, August 2008.
13. Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval.* Cambridge University Press, New York, NY, USA, 2008.
14. Filippo Menczer, Gautam Pant, and Padmini Srinivasan. Topical web crawlers: Evaluating adaptive algorithms. *ACM Transactions on Internet Technology*, 4:http://dollar.biz.ui, 2004.
15. The Lemur Project. A dataset to support research on information retrieval and related human language technologies. http://lemurproject.org/clueweb09.php, 2014.
16. Sergej Sizov, Martin Theobald, Stefan Siersdorfer, Gerhard Weikum, Jens Graupmann, Michael Biwer, and Patrick Zimmer. The bingo! system for information portal generation and expert web search. In *CIDR*, 2003.