2009

Discovering the size of a deep web data source by coverage

Jie Liang
University of Windsor

DISCOVERING THE SIZE OF A DEEP WEB DATA SOURCE BY COVERAGE

by

Jie Liang

A Thesis
Submitted to the Faculty of Graduate Studies
through the School of Computer Science
in Partial Fulfillment of the Requirements for
the Degree of Master of Science at the
University of Windsor

Windsor, Ontario, Canada
2009

DISCOVERING THE SIZE OF A DEEP WEB DATA SOURCE BY COVERAGE

by

Jie Liang

APPROVED BY:

Dr. Huapeng Wu
Department of Electrical and Computer Engineering

Dr. Luis Rueda
School of Computer Science

Dr. Jianguo Lu, Advisor
School of Computer Science

Dr. Jessica Chen, Chair of Defense
School of Computer Science

21 August 2009

Author’s Declaration of Originality

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyone’s copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis and have included copies of such copyright clearances to my appendix.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office, and that this thesis has not been submitted for a higher degree to any other University or Institution.

Abstract

The deep web is a part of the web that can only be accessed via query interfaces. Discovering the size of a deep web data source has been an important and challenging problem ever since the web emerged. The size plays an important role in crawling and extracting a deep web data source. The thesis proposes a new estimation method based on coverage to estimate the size. This method relies on the construction of a query pool that can cover most of the data source. While it is trivial to use a large dictionary such as Webster to cover the entire data source, the variance of the estimation is too large due to the large variance of the document frequencies of words. We propose two approaches to constructing a query pool so that document frequency variance is small and most of the documents can be covered. Our experiments on four data collections show that using a query pool built from a sample of the collection will result in lower bias and variance. Also, it is less costly in terms of the number of queries issued and the number of documents downloaded. In addition, we compared the new method with three existing methods based on the corpora collected by us.

To my parents and my cousin - Wen Yang.

Acknowledgements

My thanks and appreciation to Dr. Jianguo Lu for persevering with me as my advisor throughout the time it took me to complete this research and write the thesis, and for his guidance on my research and during the course of my graduate study which are the most important experiences in my life.

I am grateful as well to Dr. Richard A. Frost for his instructions on conducting a literature review and survey on the topic of my thesis. Especially, I need to express my gratitude and deep appreciation to Dr. Jessica Chen who encouraged and enlightened me on the way of critical thinking in the early stages of my graduate study and research.

I am grateful too for the support and advise from Yan Wang who is a PhD student conducting related topics to mine with my advisor. I must acknowledge as well the friends of mine who shared their memories and experiences in Canada.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The deep web [8], in contrast to the surface web that can be assessed by following hyperlinks inside web pages, consists of the resources that can only be obtained via query interfaces such as HTML forms [7] and web services [36]. Data from deep web are usually generated from background databases of websites.

The size of a data source is a vital parameter used in the collection selection algorithms in distributed information retrieval systems [41] [42] [40]. Also, estimating the size of a data source is a necessary step of a deep web data crawler and data extractor [27] [7] [35] [13] [14] [41] [32] [34] [39] [44] [20]. They need to know the size to decide when to stop crawling. In addition, the size is an important metric to evaluate the performance of the crawler and the extractor. Although the data source owner knows its size, that information may not be available to the third parties.

The objective of the thesis is to estimate the number of documents a deep web data source contains by sending queries. The estimation process starts with selecting queries. Queries can consist of characters, words, phrases, or their combinations. The queries can be randomly selected, or chosen according to their features such as their document frequencies.

After queries are issued to capture the documents in a data source, matched documents or their IDs are returned. Using this data, several methods are developed to estimate the size. In general, there are two approaches to estimating a data source size. One does not need to download and analyze the documents. Instead, it only needs the document IDs. This approach originates from a traditional *Capture-Recapture* model [38] [2] [15] [25] [17]. It analyzes the relation between the number of distinct documents and duplicates in the captures [40] [29] [30] [9] [19] [43] [5] [45] [23] [24] [21] [11]. The basic estimator has an underlying assumption: each object has equal probability to be captured. However, the capture of documents cannot be random as they can only be retrieved by queries. The other basic approach requires the downloading of the documents matched by queries [12] [6] [33] [28]. It involves document content analysis, trying to find the relations between the downloaded documents and the collection.

This thesis proposes a new coverage-based method for estimating the size of a data source. This method constructs a query pool from which queries are selected. This query pool should have high coverage of the collection but relatively low and similar document frequencies. Once the query pool is built, we will randomly select and issue a number of queries, download the documents that are captured, and compute the weight of the queries [12] to estimate the size of the collection. If query pool is not carefully constructed, this method will have negative bias, large variance, or a high cost. When the query pool cannot cover all the documents, there will be a negative bias. When the query pool contains words of very different document frequencies, there will be a large variance. When the query pool has some popular words, the cost of estimation is high. Hence, we propose two ways to construct a better query pool that can reduce the bias, variance, the number of queries, and sample size. One approach resorts to using random queries to obtain a sample of the data source. From this sample, we select queries that have similar document frequencies, and a high coverage of the data collection. The other approach assumes a collection is available to allow arbitrarily access. We utilize it, selecting a number of terms as the query pool that covers more than 99% of this collection. Then, we use this query pool to estimate other data source size.

This thesis conducts an extensive comparison between the new method and three existing estimation methods proposed by Shokouhi et al. [40], Broder et al. [12] and Lu [30], respectively. We carry out experiments to produces the data of each method under our open and flexible estimation framework. In the experiments, we collected four English and three Chinese data collections to simulate actual deep web data sources. The performance of each method is evaluated by bias, variance and cost. We issue different numbers of queries to examine the cost and bias of a method. Furthermore, we issue the same number of queries for 100 times to collect 100 estimates in order to measure the variance. In particular, we investigate the impact of sample size on the estimation accuracy. All experimental data are tabulated in tables and visualized in plots. Finally, a comprehensive comparison summary demonstrates the capabilities of each method.

This thesis is organized as follows: before proposing the new method, we summarize the related work. Researchers built different environments to evaluate their proposed methods. These environments are not open to the public. Hence, it is difficult for others to carry out experiments to compare existing methods with new ones. We create an open and flexible estimation framework to provide estimation data via web services. This framework and data collection we collected to evaluate methods are illustrated in Chapter 3. Chapter 4 describes our new approach based on Broder et al.'s concepts. In Chapter 5, we describe our improvement of Broder et al.'s method. Also, we collect evaluation data to have a comparison study of the four methods mentioned above. Finally, this thesis summarizes the advantages of each method in Chapter 6.

Chapter 2

Related Work

There are two approaches to estimating a collection size. The first approach only needs to examine the document identifiers. It originates from a traditional *Capture-Recapture* model. Methods based on this approach include the *Capture Histories with Regression* Method[40] and the OR Method[29][30]. These methods analyze the relations between stepwise overlapping of documents, historically distinct documents and totally checked documents. Because the capture of documents is not precisely random as they can only be retrieved by queries, they need to compensate for the bias introduced sampling document by queries. The second approach needs to download and analyze document. It requires the construction of query pools and the downloading of the documents response to queries. Broder et al. developed a method to measure how much a query from a pre-selected *Query Pool* contributes to capturing the documents that the Query Pool can capture in total. Furthermore, they proposed a new estimation method by employing two query pools and applying Peterson estimator [4].

2.1 The *Capture Histories with Regression* method

Shokouhi et al. [40] adapted the Capture-Recapture [38] technique used in ecology to estimate corpus size. They proposed a correction to the *Capture Histories* (*CH*) method and the *Multi Capture Recapture* method to compensate for bias inherent in sampling via query-based sampling. The authors compensated for the bias using training sets and applied regression analysis. The new methods are called the *Capture Histories with Regression* (*CH-Reg*) and *Multi Capture Recapture with Regression*.

The *CH* method issues t number of queries. After i queries, it will count the number of documents returned (k_i), the number of documents in the returned documents that have all been captured (d_i), and the total number of distinct documents that have ever been captured (u_i). The size of the collection (N) is estimated as [40]:

$$\hat{N} = \frac{\sum_{i=1}^{t} k_i u_i^2}{\sum_{i=1}^{t} d_i u_i} \tag{2.1}$$

The *CH-Reg* compensates for the bias by:

$$log(N_{CH\hat{-}Reg}) = 0.6429 \times log(N) + 1.4208 \tag{2.2}$$

There are three constraints when collecting query returns in their experiments. One is that only the top 10 documents are collected in each step. By default, a search engine has its sorting method of query returns. In this case, we use the default sorting method of Lucene. The second constraint is that any query that returns less than 20 documents is eliminated. The third constraint is that *CH-Reg* sends 5,000 queries to obtain an estimated size of a corpus.

2.2 Broder et al.'s method

Broder et al. [12] proposed the following method (Broder's method hereafter) to estimate the size of a data collection: They firstly choose two query pools. For each query pool, by issuing a random number of queries and calculating the weights of documents that can be captured by a query, and the weights of queries, it can estimate the number of documents that this query pool can capture. They estimate the overlap of two groups of documents that captured by the two query pools by using one query pool and removing the documents that contains no query in the other query pool. They estimate N by the traditional Peterson estimator [4] $\hat{N} = n_1 n_2 / n_d$ (where n_1 and n_2 are the sizes of two sets of document and n_d is the size of the intersection).

2.3 The OR Method

Lu [29][30] introduced a new term called the *overlapping rate* (OR) which is the fraction of the number of accumulative documents returned by each query to the number of unqiue documents obtained after i queries (Equation 2.4). The author approximated the relation between OR and the proportion of the corpus that k_i covered by a regression analysis. Using the Newsgroup and Reuters data collections for training, the author obtained the overlapping law in English corpora:

$$P = 1 - OR^{-1.1} \tag{2.3}$$

where

$$OR = \frac{u_i}{\sum_1^i k_i} \tag{2.4}$$

And further derived the estimator:

$$\hat{N} = \frac{u_i}{1 - OR^{-1.1}} \tag{2.5}$$

Chapter 3

The Experiment System and Data Collections

3.1 The system

Building and accessing text corpora is a necessary step in the research when estimating the size of deep web data sources. Because the size of corpora under investigation is too large to be stored in one machine, multiple machines are involved in the experiments. We propose a new open and flexible framework to share the corpora via web service so that various estimators can be experimented with. In this framework, it is easy to add new data collections, and they can be stored in different machines. The provided programmable interface would allow third parties to query existing collections. Researchers, including people in other research groups, can use their own program to configure query parameters and issue queries. The framework is able to return stepwise estimation data like k_i in Equation 2.1, and statistical data such as the query weight distribution. With the help of the proposed framework, researchers will be able to evaluate the estimation performance and result of the proposed method. It is also convenient to compare the estimation results of new methods with existing methods.

3.1.1 Overview

The framework consists of three basic components depicted in Figure 3.1.

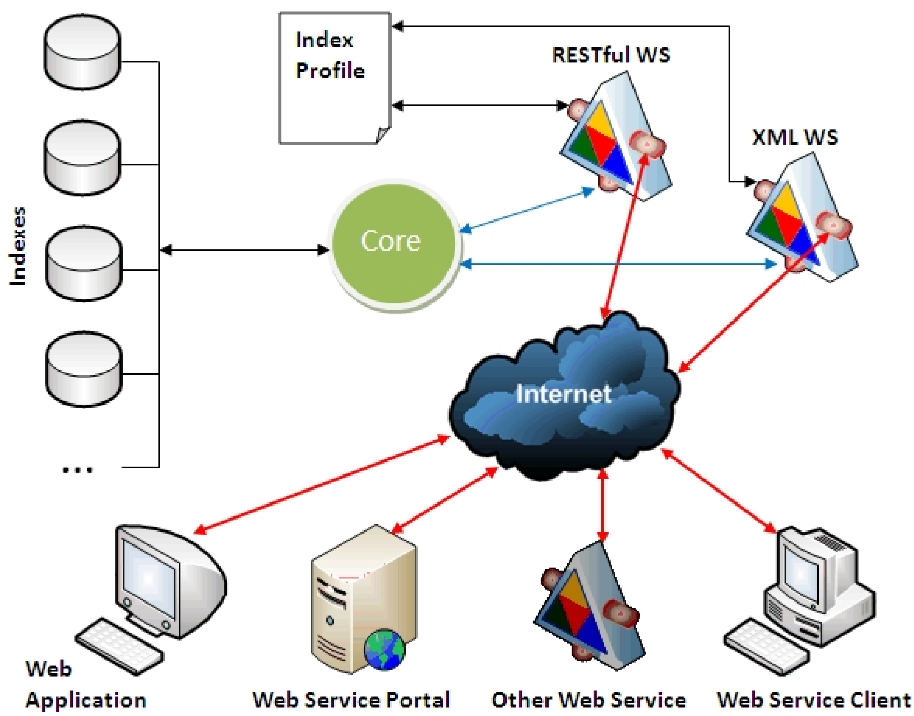

Figure 3.1: The System Overview

The estimation core is the program which accepts search arguments and queries, and returns query results and statistical data. Each invocation of the core results in a query process on one selected index. It has the predefined ability to access different type of indexes. The output remains in the same format.

We utilize two popular protocols of web service: XML Web Service and RESTful Web

Service, which have many applications [1][26]. When the request is sent to one of the web services, it will decompose the request to extract such information as the target collection, the step size, sorting approach, and terms, and invokes the estimation core with extract arguments and terms. When the core finishes a search, it will return the result to the web service. The services then send the result back to the requester. These web services could also provide the information of available collections.

The system is flexible in a way that the service could be invoked by other applications or services which wants to make further use of the data. Also, the system could be duplicated and each computer (node) could be in different machines. By a web service portal, the nodes could work together to contribute to a more powerful system.

3.1.2 Data Flow

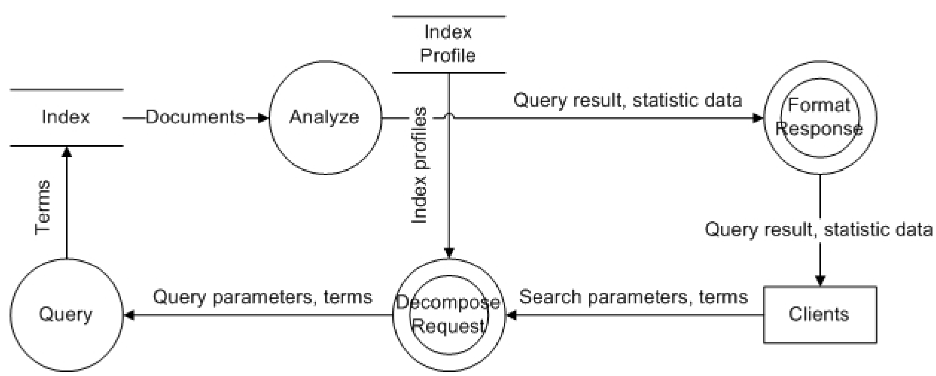

Figure 3.2: The Service Data Flow

The clients of web services send search parameters such as the name of the collection, the step size, sorting approach and queries to the web service. The web services decompose the request, and collect the profile of the desired collection; for example, the total number of documents in the index and the sorting method. Then the services invoke the core program with decomposed requests as arguments to query on the selected index. The core records the document IDs return by querying each term. If requested, it can provide various data for example u_i, d_i, the document frequency (df), the weight of each document, and the weight of each query [12]. These data are returned to web services. The web services format the query results and statistical data in XML format, and return them to clients. Note that in order to reduce network traffic, it will only return requested data.

3.2 The data collections

In our experiment, we collected seven data collections which are popular in natural language processing, information retrieval, and machine learning systems. They are TREC GOV2 Collection [16], Reuters Corpus [37], Newsgroup Corpus, English Wikipedia (enwiki-20080103-dump) and Chinese Wikipedia (zhwiki-20090116-dump) [18], a collection of Chinese literature and Sogou Web Corpus [22]. The GOV2 and Sogou Web Corpus have more than 10 million documents. In order to evaluate the methods for estimating different sizes of corpora, we created a few random subsets theses corpora. All documents are converted to UTF-8 encoding before indexing.

3.2.1 Characteristics of data collections

We collect various statistical data to provide the information of collection characteristics. Table 3.1 is a summary of the data collections. Figure 3.3 shows the size distribution of each collection. It shows that there are a few documents that have very large sizes and many documents have small sizes.

Table 3.1: The corpora summary. Cells marked by '-' mean data are not available.

Corpus	N	Mean of document size(Byte)	SD of document size	Mean of the number of unique terms	SD of the number of unique terms
Reuters	806,791	1,553	1,264	125	82
Reuters 100k	100,000	1,612	1,331	125	83
Reuters 500k	500,000	1,617	1,329	125	82
Newsgroup	1,372,911	4,582	6,177	294	223
Newsgroup 100k	100,000	4,600	6,270	295	225
Newsgroup 500k	500,000	4,575	6,180	294	222
English Wikipedia	1,475,022	4,498	6,441	284	285
English Wikipedia 100k	100,000	4,513	6,472	285	289
English Wikipedia 500k	500,000	4,482	6,438	284	286
GOV2 subset0	1,077,019	10,842	22,796	396	409
GOV2 100k	100,000	10,934	22,871	395	401
GOV2 500k	500,000	10,811	22,783	396	407
GOV2 2M	2,000,000	10,919	22,747	396	410
Chinese Wikipedia	212,042	2,771	2,721	-	-
Chinese Literature	90,749	9,512	21,069	-	-
Sogou Web Corpus 1M	1,000,000	2,348	3,633	-	-
Sogou Web Corpus 2M	2,000,000	2,372	3,960	-	-

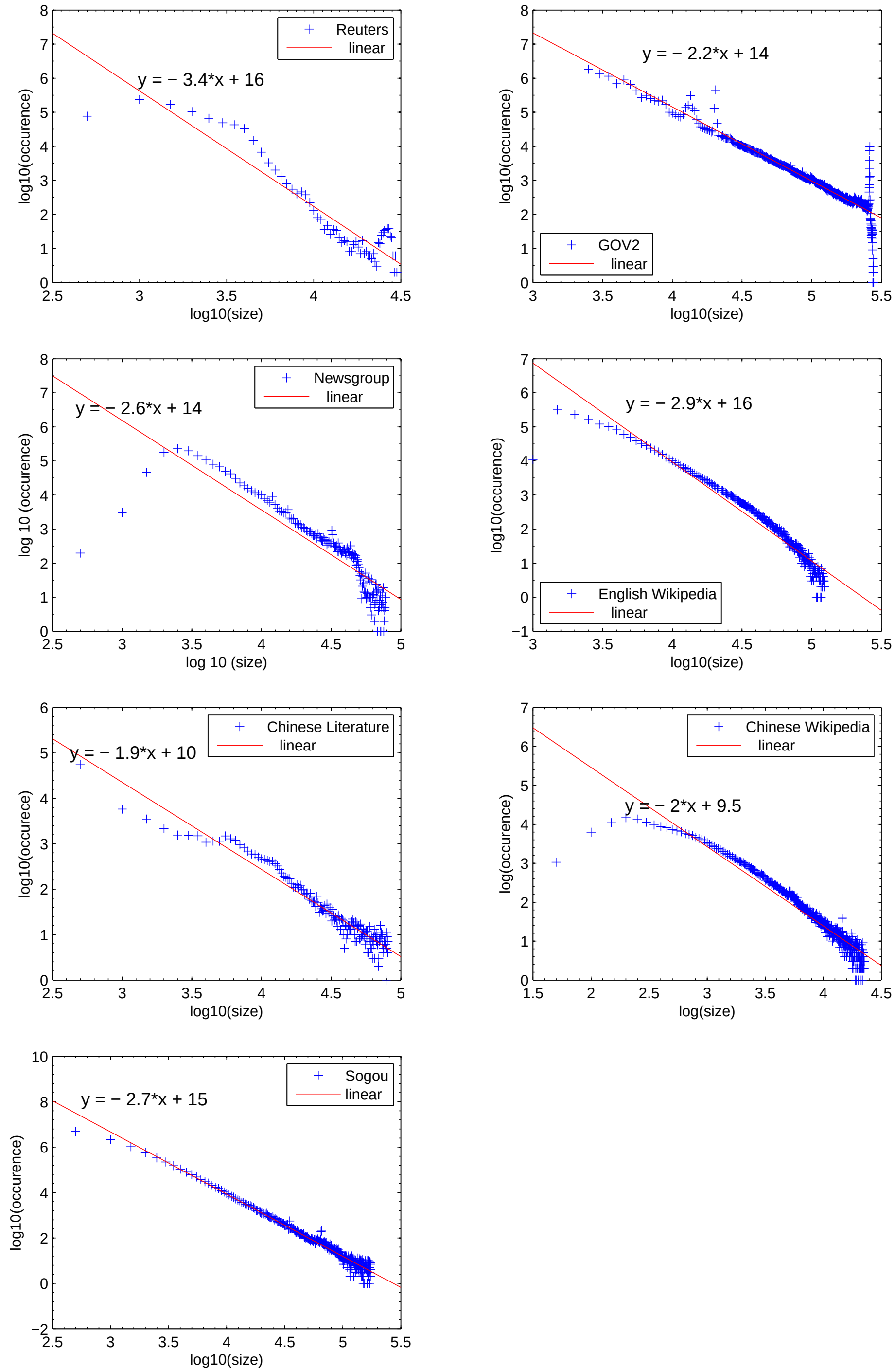

Figure 3.3: The document size distribution of all corpora.

Table 3.2: The *Field*s in each Lucene *Document* object.

Field Name	Purpose
ID	Globally identify a document.
TITLE	The title of a document.
CONTENT	The content of a document, represented by a *Term Vector*.
SIZE	The number of characters in its content that is indexed.

3.2.2 Indexing the data collections

We use Lucene [3] (2.3.0) to build the collection indexes in our experiments. Four *Fields* in each Lucene *Document* object are created. Table 3.2 shows the name of the fields and their purpose.

The *ID* field stores the file name as a document's ID. Each document has a unique ID among all collections. The query on multiple indexes can benefit from this feature when using *MultiSearcher* object in Lucene. Not all documents in the data collections have strings that can be considered as the title of the document. The content of a document is in various formats, for example, plain text, xml and html. The following lists the details of indexing different data collections:

Reuters The title of a document is obtained from "title" tag. The content is the strings under all the XML tags.

Newsgroup The title of a document is not specified in the file. We use the document ID as its title.

Wikipedia The title of a document is obtained from "title" tag. The content is the strings under all the XML tags. Redirection documents are removed.

GOV2 The title of a document is obtained from "title" tag. The content is the strings under all the HTML tags.

Sogou Web Corpus The *docno* tag is mapped to the title. The *doc* tag is mapped to the content.

3.3 Evaluation metrics

The methods introduced in the next chapters will be evaluated in terms of Relative Bias, Relative Standard Deviation and Mean Squared Error. N is estimated under the same conditions for m times (named *trials* hereafter). Let $\hat{n}_i$ denotes an estimated size by the i-th trial ($1 \leq i \leq m$). The expected value of $\hat{N}$, denoted by $E(\hat{N})$, is the mean of m estimates:

$$E(\hat{N}) = \frac{1}{m}\sum_{i=1}^{m}\hat{n}_i$$

The Relative Bias measures how close $E(\hat{N})$ to the actual size N:

$$RB = \frac{E(\hat{N}) - N}{N}$$

If RB is negative, it is underestimated. Otherwise, it is overestimated.

As a measure of precision, the Relative Standard Deviation represents how far the estimations are from the mean:

$$RSD = \frac{1}{E(\hat{N})}\sqrt{\frac{1}{m-1}\sum_{i=1}^{m}(\hat{n}_i - E(\hat{N}))^2}$$

The bias and variance can be combined using the Mean Squared Error (MSE) which defined as:

$$MSE = \frac{1}{m-1}\sum_{i=1}^{m}(\hat{n}_i - E(\hat{N}))^2 + (E(\hat{N}) - N)^2$$

Another metric of the experiments is the cost of the estimation, which is the sample size, i.e., the total number of documents checked. In general, the estimation accuracy increases when the sample size becomes larger. However, a very large sample size will make the estimation process inefficient, and reduce the estimation problem to a trivial one by downloading and counting *all* the documents. Hence, the estimation cost is an important metric when evaluating the methods.

Chapter 4

The Pool-based Coverage Method

This chapter proposes a new coverage-based method to estimate the size of a data source. We first introduce the weight of a query and an unbiased estimator. Using a dictionary to be the query pool leads an estimate to be costly. We propose two approaches of constructing a query pool that can induce large variance and high cost. The queries in a query pool are selected either from a sample of the targeted data source, or from another existing data collection. These queries should have low document frequencies and high coverage of the data source. We carry out experiments to compare which approach has the better performance measured by the metrics introduced in Section 3.3.

4.1 A naive estimator

Given a collection of documents (*D*) and a query pool (*QP*), we want to know how many documents in *D* that a query pool can match. We assume that the number of queries in *QP* is very large, hence sending all of the queries in *QP* is too expensive to be considered an option. A solution to this problem is probing the deep web using a subset of *QP*. Let us start the discussion with a simplified example. Let $QP = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8, q_9\}$ and $D = \{d_0, d_1, d_2, d_3, d_4, d_5, d_6, d_7, d_8, d_9, d_{10}\}$. The relationship between D and QP is $d_0 = \{q_0, q_1\}$, $d_1 = \{q_1\}$, $d_2 = \{q_2\}$, $d_3 = \{q_3\}$, $d_4 = \{q_4\}$, $d_5 = \{q_5\}$, $d_6 = \{q_6\}$, $d_7 = \{q_7\}$, $d_8 = \{\}$, $d_9 = \{\}$. This relationship can be represented using the query-document 10×11 matrix as in Table 4.1, where the cell having value 1 indicates that the corresponding document contains the query.

Table 4.1: The matrix represents a set of queries with the documents that they can capture. Cells marked by '1' mean a document match a query.

	d_0	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}	df	weight
q_0	1										1	$1+1$	$1/2+1$
q_1	1	1										$1+1$	$1/2+1$
q_2			1									1	1
q_3				1								1	1
q_4					1							1	1
q_5						1						1	1
q_6							1					1	1
q_7								1				1	1
q_8												0	0
q_9												0	0

Let $M(q,D)$ denote the set of documents in D that matches q. The set of documents that all the queries in QP can match is denoted by $M(QP,D)$:

$$M(QP,D) = \bigcup_{q \in QP} M(q,D)$$

For example, in Table 4.1, then $M(QP,D) = \{d_0, d_1, d_2, d_3, d_4, d_5, d_6, d_7, d_{10}\}$.

If we sum the df of all the queries, it is greater than $|M(QP,D)|$ because d_0 is captured twice. However, if we allocate the total count of d_0 by distributing 1/2 to each query that d_0 contains, count the total *weight*s instead of dfs, then the sum of weights equals to

$|M(QP,D)|$.

Let us generalize the scheme of counting the $|M(QP,D)|$ in order to introduce the weight of a query: If a document contains t queries from a query pool, when querying by the query pool, d is captured t times. However, we only want d contributes 1 to keep the sum of weights equals to $|M(QP,D)|$. Hence, we can distribute $1/t$ to each of the t queries that matches d. The $1/t$ is defined as the weight of a document w.r.t. a QP. And the 'weight' used in Table 4.1 is called the weight of a query w.r.t. a QP.

The *weight of a document w.r.t. a QP* is the reciprocal of the number of queries in the query pool QP that d contains.

$$w(d,QP) = \frac{1}{|d \cap QP|} \tag{4.1}$$

For example, as demonstrated in Table 4.1, $w(d_0,QP) = \frac{1}{2}$.

The *weight of a query w.r.t. a QP* is the sum of the weights of all the documents containing q:

$$w(q,QP) = \sum_{q \in d, d \in M(QP,D)} w(d,QP) \tag{4.2}$$

For example, as demonstrated in Table 4.1, $w(q_1,QP) = w(d_0,QP) + w(d_1,QP) = \frac{1}{2} + \frac{1}{1} = \frac{3}{2}$.

The *weight of a query pool* is the average of the weights of all the queries in the QP.

$$W(QP,D) = \frac{\sum_{q \in QP} w(q,QP)}{|QP|} \tag{4.3}$$

In the example shown in Table 4.1, $W(QP,D) = 0.9$.

Broder et al. proposed an *Unbiased Estimator* to estimate $|M(QP,D)|$. They proved that

$|M(QP,D)|$ equals to $W(QP,D)$ multiplied by $|QP|$:

$$|M(QP,D)| = W(QP,D) \times |QP| \tag{4.4}$$

In the real estimation, we apply the Simple Random Sampling with Replacement technique: In each estimation, we randomly select t number of queries from the query pool, issue them and download the documents that matches the queries. Note that the calculation of the weight of a query does not need to issue all the queries in the query pool. Then we can calculate the average weight of t queries. This is an estimated value of $W(QP,D)$. And we know $|QP|$ because it is selected. So the estimated size is $|QP|$ multiplies by the estimated average weight.

We use the example shown in Table 4.1 to demonstrate how to estimate $|M(QP,D)|$. We estimate the size by 2 trials to show how to obtain RSD and MSE. Each trial randomly selects two queries.

Estimation 1: q_1 and q_5 are selected. We issue them, d_0, d_1 and d_5 are downloaded. The document weights are calculated: $w(d_0,QP) = 1/2$, $w(d_1,QP) = 1/1$, $w(d_0,QP) = 1/1$. And the query weights can be calculated: $w(q_1,QP) = w(d_0,QP) + w(d_1,QP) = 3/2$, $w(q_5,QP) = w(d_5,QP) = 1$. The average weight is $5/4$. Then the estimated size is $|M(\hat{QP},D)|_1 = 5/4 \times 10 = 12$.

Estimation 2: q_3 and q_6 are selected. We issue them, d_3 and d_6 are downloaded. The document weights are calculated: $w(d_3,QP) = 1$, $w(d_6,QP) = 1$. And the query weights can be calculated: $w(q_3,QP) = w(d_3,QP) = 1$, $w(q_6,QP) = w(d_6,QP) = 1$. The average weight is 1. Then the estimated size is $|M(\hat{QP},D)|_2 = 1 \times 10 = 10$.

Now we can calculate $E(|M(\hat{QP},D)|) = (12+10)/2 = 11$, $RB = (11-9)/9 = 2/9$, $RSD = 0.1286$ and $MSE = 6$.

There is still a gap between $|M(QP,D)|$ and N. Broder et al. estimated N by the Peterson estimator [12]. We propose a new one that needs to build a query pool to cover most of documents in D, if QP is constructed appropriately. In this case, $|M(QP,D)| \approx N$. More precisely, let Dic denote a dictionary, we want to estimate N that is defined as:

$$N = |\bigcup_{q \in Dic} M(q,D)| \tag{4.5}$$

When $QP = Dic$, an unbiased estimator for N is given in Algorithm 1, which is borrowed from Broder et al.'s estimation method by query weight [12].

Algorithm 1: The Coverage based estimation algorithm

Input: A query pool QP, the number of queries t to be sampled, a data collection D.
Output: Estimate $\hat{N}$.

1. Randomly select t number of queries $q_1, q_2, \ldots, q_t$ from QP, let $random(t, QP)$ denote the set of queries selected.

2. Send the queries to D and download all the matched documents.

3. For each $q \in random(t, QP)$, calculate $w(q, QP)$.

4. $\hat{N} = \frac{\sum_{i=1}^{t} w(q_i, QP)}{t} |QP|$.

In many cases, we cannot use words from Dic as queries directly. For example, when a data source is large we need to use conjunctive queries consisting of multiple words from Dic so that the return set is not too large. Additionally, words within a certain df range are preferred. Hence, there is a need to construct a query pool that can contain single words or multi words queries built from Dic.

4.2 The large variance and high cost problem

If we run Algorithm 1 with the query pool that equals to the Webster Dictionary, the estimator has no bias as Broder et al. have shown in [12]. However, the variance could be very large which renders the method impractical. Therefore, we need to construct an appropriate query pool so that the queries have similar query weights and the query pool can match almost all the documents in a data source. In this section, we show that if the query pool is selected randomly from a dictionary, such as the Webster Dictionary, the estimation will have a very large variance, as expected. We denote this approach by C1. Then we discuss two ways to construct a query pool in the next section.

The query pool we are using in this section consists of 40,000 words from the Webster Dictionary. Table 4.2 provides the statistical data of this query pool on the four collections. In our experiments, t ranges between 50 and 1000. We run 100 trials to measure the variance of an estimate by t number of queries. Each trial is independent. The experiment results are tabulated in Table 4.3.

Table 4.2: The statistical data of the query pool of C1.

	Reuters	**GOV2 subset0**	**Newsgroup**	**English Wikipedia**
max df	806791	696878	1165272	1066195
min df	0	0	0	0
mean df	1283	4399	5770	4511
df RSD	8.5555	5.5950	6.2986	5.5338
max weight	22003.45	8724.58	8967.26	12202.55
min weight	0	0	0	0
W(QP,D)	20.1703	26.9230	34.3247	36.8789
weight SD	221.3443	190.3310	237.2438	236.2848
weight RSD	10.9738	7.0695	6.9118	6.4069

The mean and the RSD of the weights of t randomly sampled queries can be derived from Table 4.2 using the Central Limit Theorem. Because Algorithm 1 applies the random sampling with replacement technique, the expected mean weight of t randomly selected queries is equal to $W(QP, D)$. If all possible samples of t number of queries are taken, according to the Central Limit Theorem, we have [10]:

$$SD_t = \frac{SD}{\sqrt{t}}$$

and

$$RSD_t = \sqrt{\frac{RSD^2}{t}} \tag{4.6}$$

In our experiments, we try 100 trials. Although 100 trials are small portions of all possible samples for each t number of queries, we can roughly know what is the RSD of query weights when we randomly sample t number of queries.

Table 4.3: The estimation by C1 on four English corpora. Data are obtained by 100 trials, each trial is produced by randomly selecting t number of queries from QP.

Corpus	Metric	t				
		50	100	200	500	1000
Reuters	mean n	59,121	139,608	251,532	697,578	1,263,021
	RB	-0.1332	0.0678	0.0099	0.0960	-0.0223
	RSD	1.1236	0.8097	1.1305	0.4969	0.3535
GOV2 subset0	mean n	246,010	466,379	842,292	2,155,689	4,428,266
	RB	0.1620	0.0351	-0.0039	-0.0231	0.0148
	RSD	1.0415	0.7572	0.5204	0.3199	0.2432
Newsgroup	mean n	279,561	565,625	1,154,411	3,054,976	5,996,794
	RB	-0.0357	-0.0285	-0.0011	0.0620	0.0396
	RSD	0.9047	0.6837	0.4850	0.3281	0.1805
English Wikipedia	mean n	268,234	449,604	876,602	2,208,401	4,519,216
	RB	0.2462	-0.0256	-0.0543	-0.0237	-0.0005
	RSD	1.0415	0.5688	0.4084	0.2881	0.2205

Table 4.3 shows that using random queries has a low bias as expected, even if only 50

queries are issued. However, the variance is too large to be of practical application. For example, if we obtain 100 estimations of GOV2 subset0 collection with each estimation uses 50 random Webster words, RSD=1.0415. We plot the estimated sizes in Figure 4.1. In this figure, the red line is the average of 100 estimates. The average estimated size of GOV2 subset0 collection is 1,251,629. However, an estimate could reach 67,338 or 8,430,879. In English Wikipedia plot, the RSD=0.2205, it can be seen from the plot that the estimated sizes are gathered relatively in a much more narrow area. Although using 1000 words to estimate the size English Wikipedia corpus has lower variance, the cost is high according to Table 4.3.

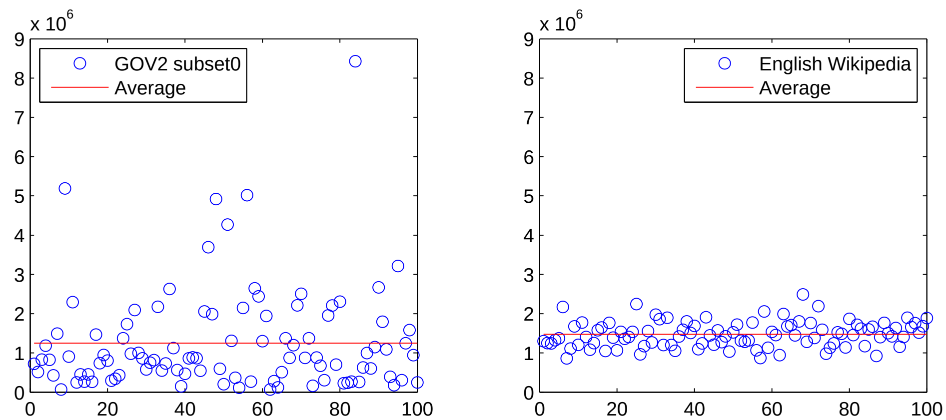

Figure 4.1: Scatter plot of 100 estimations method C1. For GOV2 subset0, 100 words are randomly selected from the 40,000 Webster words, RSD=1.0415. For English Wikipedia, 1000 words are randomly selected from the 40,000 Webster words, RSD=0.2205.

The large variance is caused by the variation of the query weights. Figure 4.2 depicts the weight distribution over *df* of the corpora under investigation. It shows that queries with similar *df*s have small variations in query weights.

Based on this observation, Broder et al. proposed to construct one of the query pools by the words having medium/low *df*s. However, rare words will not be able to cover all

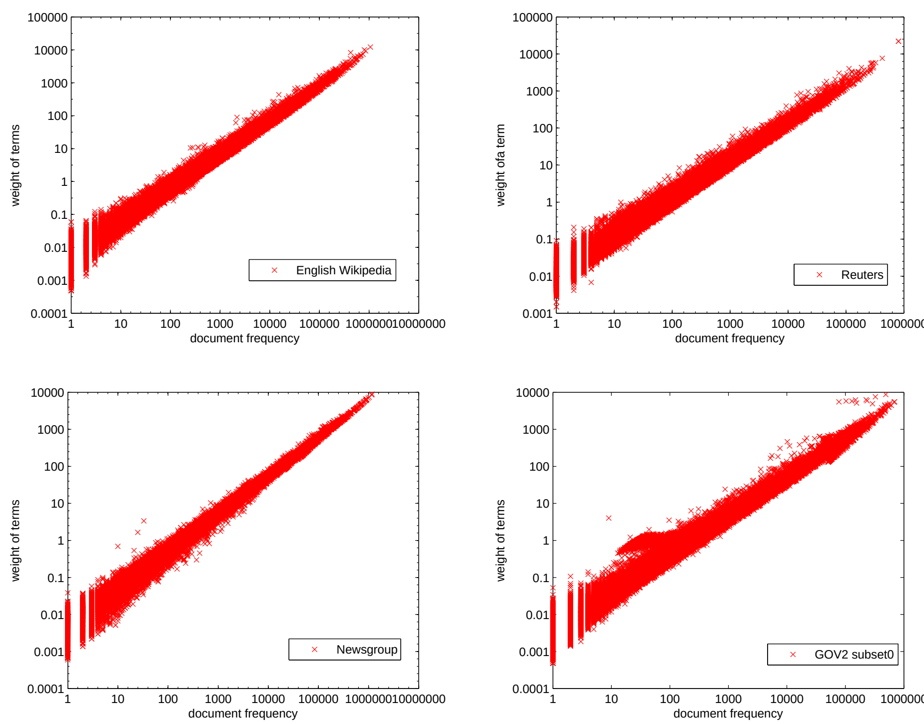

Figure 4.2: Scatter plot of query weight over *df* on the four corpora.

the documents as demonstrated in Table 4.12. In addition, they extracted all the terms from a collection to calculate *df*. Although this approach works well when the entire data collection is available for lexical analysis, it is not possible to obtain such knowledge of *df* when estimating a data collection with a query interface. Hence, we need to learn the *df* information from other sources.

4.3 Constructing a query pool

We propose two ways to construct a query pool, either from another existing corpus that is completely available to download, or from a sample of the collection whose size is being estimated.

More formally, given a data collection D', which can be a sample of data collection D, or some other data collections, our task is to construct a set of terms from D' such that:

1. The queries in the query pool should have low *df* in data collection D.

2. The queries in the query pool should cover most of the documents in data collection D.

Subgoals 1) and 2) contradict each other. When the *df* is low, it is not easy to cover most of the documents. Table 4.12 demonstrated this problem. Contrary to this, it will be too costly to use a query pool with high *df* queries. We need to select an appropriate *df* range so that both 1) and 2) can be satisfied. In the following sections, we show how we construct query pools from a corpus that is available to download, and from a sample of the collection.

In general, there are three parameters should be considered when constructing a query pool:

The size of the sample of D We reported that a sample of 3,000 documents is good enough to represent D in [31]. However, it can be changed based on different situations.

Starting df Using queries with low df will guarantee low variance of query weights and n. In the real application, it could start with the lowest df to collect queries from a sample if the number of queries is not an important factor.

Coverage of the sample Coverage of the sample directly relates to the bias of an estimate. There is no fixed relation between them reported. In the real application of this method, if 0.02 of RB is acceptable, then the coverage of the sample could be set to 98%.

In our experiments, we choose different settings of the parameters when building a query pool. These settings are denoted in Table 4.4, which are explained in the later sections. The terms we extracted from a document are tokenized by Lucene using its default *StandardAnalyzer* class.

Table 4.4: The notations of Pool-based Coverage Method. The sample size is 3000 for C3X.

Notation	**Approach**	**Settings**
C1	Using random words from Webster	-
C2	Using low frequency terms from a corpus	$df \geq 1$,coverage=0.995
C31	Using low frequency terms from a sample	$df \geq 2$,coverage=0.995
C32		$df \geq 1$,coverage=0.95

4.3.1 Learning the query pool from another existing corpus (C2)

In this section we study the approach that learns the queries from another existing data collection. For example, when estimating the size of the Wikipedia corpus, we construct the query pool by Reuters corpus. This approach is denoted by C2. The construction starts with extracting all the terms from Retuers corpus and sorts them by their *df*s in ascending order. We eliminate terms with $df = 1$ and start to collect terms until they can cover 99.5% of Reuters corpus. There are 40,340 queries selected from the terms of Rueters corpus. Table 4.5 records the information of this query pool. Comparing the query pools in C1 and C2 as tabulated in Table 4.2 and Table 4.5, we can see that the learning process of C2 reduces the average number of document the queries can match, which is represented by the 'mean df'. But the variance of weights increases. This is because there are a large number of common words in Webster. Their weights are in a more narrow range. The queries selected by C2 need to cover 99.5% of Retuers. Hence, their overall weight range is larger. This also indicates that the estimation by C32 will not have a lower variance than C1 according to the statistical data.

Table 4.5: The statistical data of the query pool for C2. The query pool has 40,340 queries.

	Reuters	**GOV2 subset0**	**Newsgroup**	**English Wikipedia**
Coverage of D	99.5%	99.9%	99.9%	99.9%
max df	185570	474991	882129	691172
min df	1	0	0	0
mean df	217	879	911	827
df RSD	12.9602	10.0205	13.6673	11.7910
max weight	26486.88	20681.30	44009.47	48981.93
min weight	0.0010	0	0	0
W(QP,D)	19.9067	26.6720	34.0120	36.5499
weight SD	281.9400	315.0866	520.7704	511.6770
weight RSD	14.2345	11.8137	15.3114	13.9994

When the query pool is built, we use it to run Algorithm 1 on four English corpora. Table 4.6 shows the estimation result. It indicates that this approach produces similar accuracy to C1 and its cost is lower than C1. For example, C1 needs to check less than 200k docu-

Table 4.6: The estimation by C2 on five English corpora. Data are obtained by 100 trials, each trial is produced by randomly selecting t number of queries from QP.

Corpus	Metric	t				
		50	100	200	500	1000
Reuters	mean n	11,240	22,246	46,167	101,886	219,259
	RB	0.0382	-0.0055	0.0461	-0.0607	0.0008
	RSD	1.8073	1.3852	0.9833	0.5944	0.5351
GOV2 subset0	mean n	39,568	79,107	172,080	383,080	823,430
	RB	-0.1150	-0.1113	0.0189	-0.1539	-0.0855
	RSD	2.0989	1.3048	0.8247	0.4918	0.3780
Newsgroup	mean n	41,378	106,790	174,700	415,589	958,693
	RB	-0.1099	0.2167	-0.0329	-0.1027	0.0655
	RSD	2.0868	1.9520	1.2049	0.6151	0.4946
English Wikipedia	mean n	49,972	81,719	163,632	411,691	823,085
	RB	0.2413	-0.0195	0.0062	-0.0118	0.0070
	RSD	1.9624	1.3567	1.0874	0.6930	0.5292

ments to produce RB=-0.0329 when estimating the size of Newsgroup collection. However, C1 needs to check around 300k documents to achieve similar RB.

We visualize the data from Table 4.3 and Table 4.6 in Figure 4.3. It shows that C2 can produce low MSE when the cost is similar to C1 in general. Since C1 produces large variance and high cost as shown in Table 4.3, the use of random words from a dictionary should be discarded in this method.

4.3.2 Learning the query pool from a sample of D (C3)

In this section, we explore learning a query pool from a sample of D. In section 4.3 we discuss there are three parameters need to be considered when learning a query pool from a sample of D. The constructing process is demonstrated in Algorithm 2.

We choose two settings of the parameters which are denoted by C31 and C32. In [31] we discussed that in a D' which has 3000 random documents, the terms whose df ranged

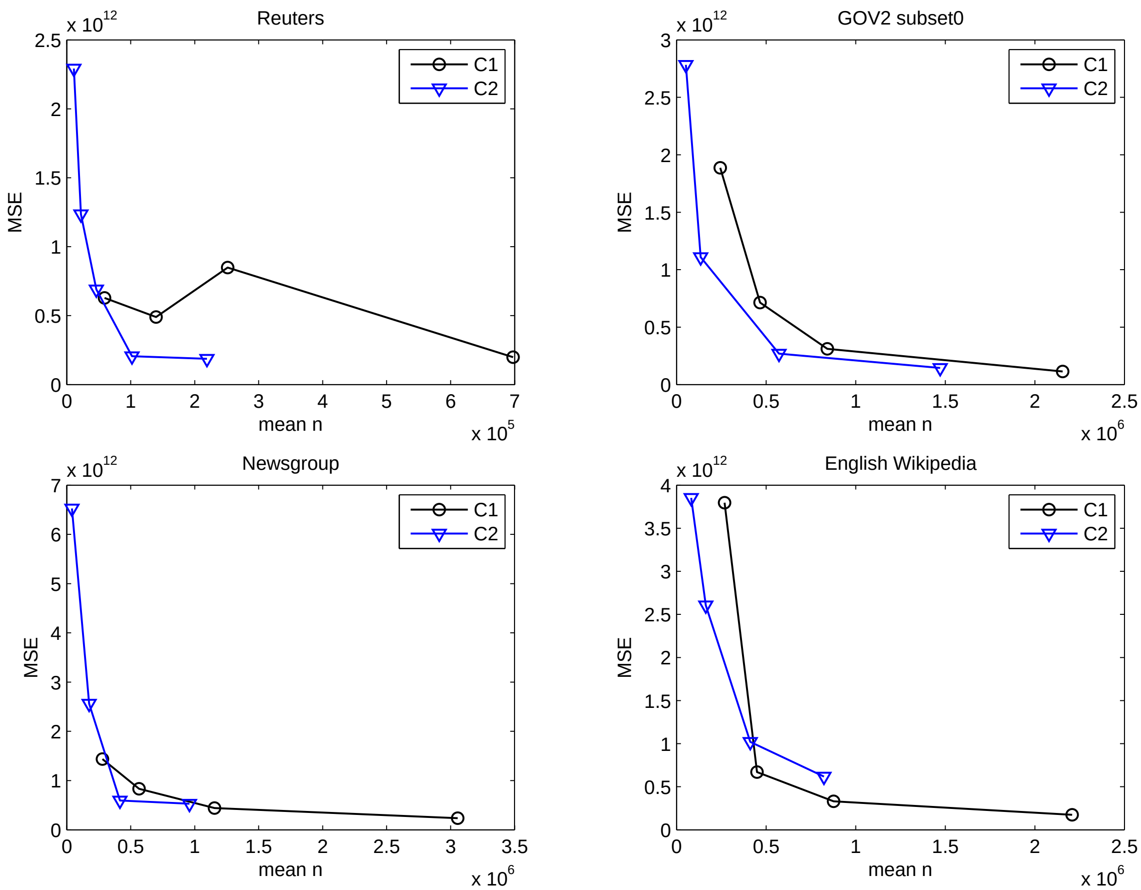

Figure 4.3: A comparison between C1 and C2. The data are from Table 4.3 and Table 4.6.

Algorithm 2: The construction of a query pool by C3

Input: A dictionary *Dic*, a data collection D, the size of the sample s, the starting document frequency df_{init} and the coverage of the sample p.
Output: Construct QP.

1. Randomly select a word from *Dic*;
2. Send the word to D and download all the matched documents to D'.
3. Repeat 1 and 2 until s number of documents are downloaded.
4. Extract all the terms in D' and sort them by df in D' in ascending order.
5. Remove the terms with $df < df_{init}$.
6. Start with the first one, collect and send the terms one by one to D' and collect all the matched document IDs until $p \times s$ distinct IDs are collected.
7. Save all the collected terms in QP.

from 2 to $0.2 \times |D'|$ could cover 99.5% of D. Hence, we set $s = 3000$ for C3.

C31

We choose $df_{init} = 2$ and $p = 99.5\%$ for the initial settings of C3. The size of the QP for each of the corpora and the QP's coverage of D is recorded in Table 4.7. It also has other information of the query pools for four collections.

From the statistical data in Table 4.7 and Table 4.5 we can see that queries selected by C31 have much smaller RSD of query weights than that of C2. This indicates learning queries from a sample can reduce the variance of estimation.

We run Algorithm 1 using the query pool constructed by C31 on four English corpora. The results are shown in Table 4.8.

Comparing Table 4.8 with Table 4.3, we can see that using low frequency terms from a sample of D significantly reduces the cost. When t is less than 1000 queries, it only needs

Table 4.7: The statistical data of the query pool of C31.

	Reuters	**GOV2 subset0**	**Newsgroup**	**English Wikipedia**
$\lvert QP \rvert$	10,847	74,399	17,016	38,853
Coverage of D	98.90%	99.89%	99.10%	99.01%
max df	20676	82144	66243	31228
min df	2	2	2	2
mean df	951	147	929	517
df RSD	1.3219	0.6440	2.5506	0.1358
max weight	4339.4	6781.75	15171.15	3122.23
min weight	0.02	0.001	0.001	0.001
W(QP,D)	73.6374	14.4190	79.9570	37.5847
weight SD	122.9127	189.2918	225.0074	70.1633
weight RSD	1.6691	13.1279	2.8141	1.8668

Table 4.8: The estimation by C31 on four English corpora. Data are obtained by 100 trials, each trial is produced by randomly selecting t number of queries from QP.

Corpus	Metric	t				
		50	100	200	500	1000
Reuters	mean n	46,916	96,373	191,286	474,995	950,882
	RB	0.0056	0.0090	0.0000	-0.0088	-0.0054
	RSD	0.2710	0.1792	0.1301	0.0657	0.0500
GOV2 subset0	mean n	16,958	30,725	67,010	171,088	328,114
	RB	0.0692	0.0335	-0.0005	-0.0280	0.0096
	RSD	1.4253	1.5558	0.6777	0.5199	0.4505
Newsgroup	mean n	48,837	93,953	185,789	461,647	926,226
	RB	0.0786	0.0233	-0.0137	-0.0145	-0.0165
	RSD	0.5470	0.3289	0.1948	0.1251	0.0928
English Wikipedia	mean n	25,596	52,351	102,003	259,146	517,210
	RB	0.0000	0.0064	-0.0216	-0.0019	-0.0069
	RSD	0.2868	0.1981	0.1363	0.0847	0.0589

to check around 500 thousand documents, while 100 queries have already resulted in more than 700 thousand documents being checked for three corpora using randomly words. Also, the variance declined as much as 10 times except for the GOV2 subset0 collection. C31 has successfully reduced the variance of weights except for GOV2 subset0. Take Reuters for instance, the RSD of weights of random words for C1 is 10.9738, the RSD of the query pool selected by C31 is 1.6691. The less variance of a query pool, the smaller variation of the estimation.

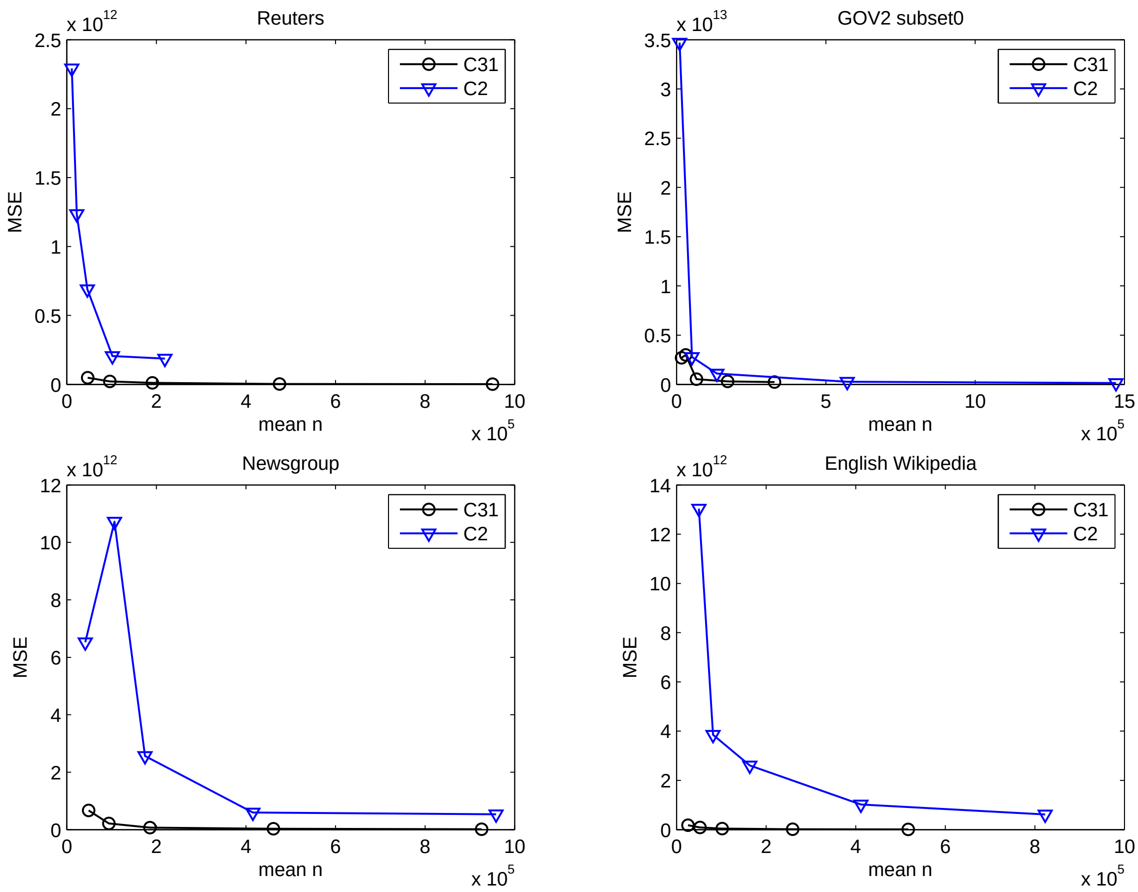

Figure 4.4: A comparison between C2 and C31. The data are from Table 4.6 and Table 4.8.

We visualize the data from Table 4.6 and Table 4.8 in Figure 4.4. It shows that C31 is able to produce lower bias and smaller variance than C2.

C32

We examine if the cost of C31 could be further reduced by choosing another set of values for the parameters. Although C31 can produce RB< 0.1 for almost all corpora, the cost is high according to Table 4.8. For example, using 500 queries to estimate the size of Reuters corpus, it needs to check around 2/3 of the documents that Reuters has. Therefore, we decrease p to 95%. As the coverage is changed, in order to maintain low variance, we change the df_{init} from 2 to 1. We build a query pool for each corpus using above the settings.

Table 4.9 presents the information of the query pools for four collections. Figure 4.6 and Figure 4.7 present the $weight - df$ distribution of queries selected by C32 and random words. They show that dfs and weights of queries learnt by C32 are in a more narrow ranges than those of random words. It means the cost and variance are successfully reduced. By comparing the statistical data of the query pools for C31 and C32, we can observe that the cost can be further reduced by C32 except for Newsgroup corpus. However, the variance of estimation can not be lower because the RSD of weights are higher than those of C31 for almost all collection. The experimental data of C32 are recorded in Table 4.10 and Table 4.11.

C32 tries to further reduce the cost of C31 by sacrificing some accuracy. Figure 4.5 depicts the performance of C31 and C32.

Figure 4.5 proves that C32 successfully reduces the cost when estimating Reuters, GOV2 subset0 and English Wikipeida. And it is able to produce lower bias and variance than C31. However, when C32 estimates the size of English Wikipedia corpus, the bias is larger than C31, although the cost is much less.

Table 4.9: The statistical data of the query pool of C32.

	Reuters	**GOV2 subset0**	**Newsgroup**	**English Wikipedia**
$\|QP\|$	20,862	117,101	40,038	45,449
Coverage of D	95.60%	94.68%	99.8%	84.76%
max df	24758	51432	119457	6821
min df	1	1	1	1
mean df	228	79	1042	77
df RSD	2.9443	1.8635	2.3966	0.8477
max weight	7799.26	16913.84	2325.06	2387.61
min weight	0.01	0.001	0.0006	0.002
W(QP,D)	36.9721	8.6577	14.6051	23.3278
weight SD	139.6667	185.5397	33.4347	93.1826
weight RSD	3.7776	21.4305	2.2892	3.9945

Table 4.10: The estimation by C32 on five large English corpora. Data are obtained by 100 trials, each trial is produced by randomly selecting t number of queries from QP.

Corpus	Metric	t				
		50	100	200	500	1000
Reuters	mean n	9,386	19,168	39,061	94,626	193,303
	RB	-0.0747	-0.0455	-0.0367	-0.0641	-0.0415
	RSD	0.3516	0.2783	0.1983	0.1068	0.0783
GOV2 subset0	mean n	3,991	7,738	15,354	39,366	81,233
	RB	-0.2442	-0.2035	-0.2008	-0.1779	-0.1635
	RSD	1.6602	1.4977	1.1166	0.8292	0.5016
Newsgroup	mean n	28,085	56,773	114,207	292,097	584,513
	RB	-0.0692	-0.0870	-0.0761	-0.0568	-0.0647
	RSD	0.6953	0.2812	0.3161	0.1820	0.1104
English Wikipedia	mean n	3,517	7,046	14,268	36,005	71,435
	RB	-0.2546	-0.2440	-0.2367	-0.2339	-0.2378
	RSD	0.4846	0.3404	0.2179	0.1352	0.0894

Table 4.11: The estimation by C32 on small English corpora. Data are obtained by 100 trials, each trial is produced by randomly selecting *t* number of queries from *QP*.

Corpus	Metric	t				
		50	100	200	500	1000
Reuters 100k	mean n	832	1,634	3,301	8,218	16,496
	RB	-0.0616	-0.0762	-0.0704	-0.0653	-0.0660
	RSD	0.3212	0.2508	0.1643	0.1030	0.0684
Reuters 500k	mean n	5,020	10,264	20,574	51,120	101,517
	RB	-0.0770	-0.0487	-0.0434	-0.0649	-0.0740
	RSD	0.4046	0.3301	0.2168	0.1072	0.0734
GOV2 100k	mean n	361	712	1,482	3,425	7,213
	RB	-0.1998	0.0685	0.0157	-0.0940	-0.0474
	RSD	2.1007	2.3951	1.3711	0.8929	0.7579
GOV2 500k	mean n	3,137	7,416	12,280	33,255	66,865
	RB	0.0630	0.1117	-0.0136	-0.1291	-0.0641
	RSD	4.0719	2.0461	1.7858	0.5664	0.5695
Newsgroup 100k	mean n	1,585	3,013	6,119	15,551	30,788
	RB	-0.0484	-0.0936	-0.0939	-0.0648	-0.0732
	RSD	0.4172	0.2625	0.1934	0.1332	0.1045
Newsgroup 500k	mean n	9,454	17,749	35,446	90,122	184,335
	RB	-0.0613	-0.0907	-0.0910	-0.0873	-0.0609
	RSD	0.3489	0.2417	0.2142	0.1378	0.0893
English Wikipedia 100k	mean n	1,023	2,086	3,923	1,003	18,356
	RB	0.1095	-0.2307	-0.1882	-0.1492	-0.1812
	RSD	0.4216	0.3948	0.1877	0.1582	0.0884
English Wikipedia 500k	mean n	903	2,549	4,184	9,045	20,375
	RB	-0.3023	-0.1891	-0.2589	-0.2850	-0.3051
	RSD	0.6134	0.6483	0.5893	0.2497	0.1538

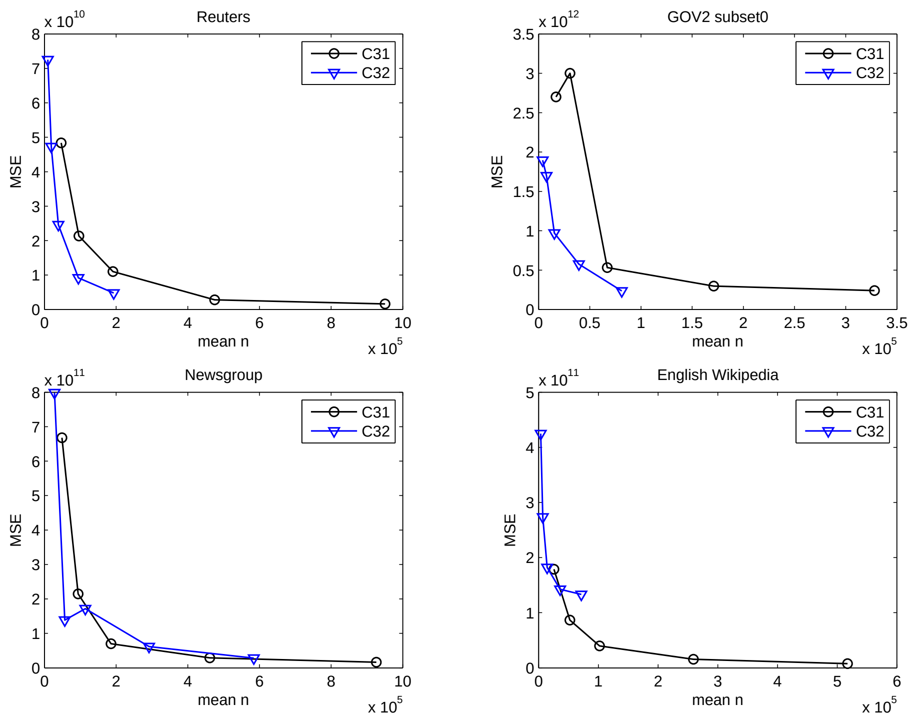

Figure 4.5: A comparison between C31 and C32. The data are from Table 4.8 and Table 4.10.

4.4 Summary

Generally, Figure 4.3, Figure 4.4 and Figure 4.5 show that C3 is more cost effective and C2 and C1 because it produces lower MSE and checks less documents. And C32 is slightly better than C31 in terms of the cost. The parameters make this method adjustable and flexible to estimate the size of an actual deep web data source. As we proved, using a sample of D to collect terms will result in the best estimation. In Chapter 5, we compare C32 with the other three existing methods.

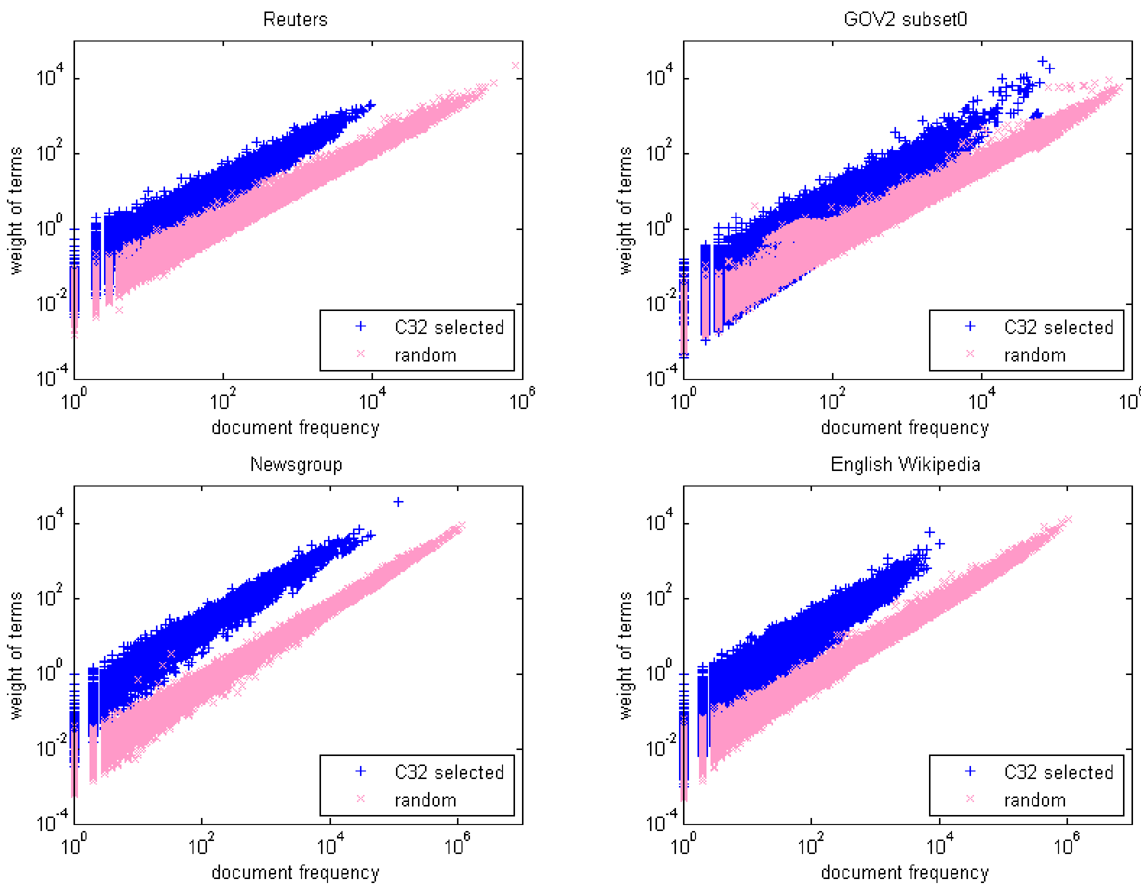

Figure 4.6: Weight of terms from a sample of D and weight of random words distribution over df.

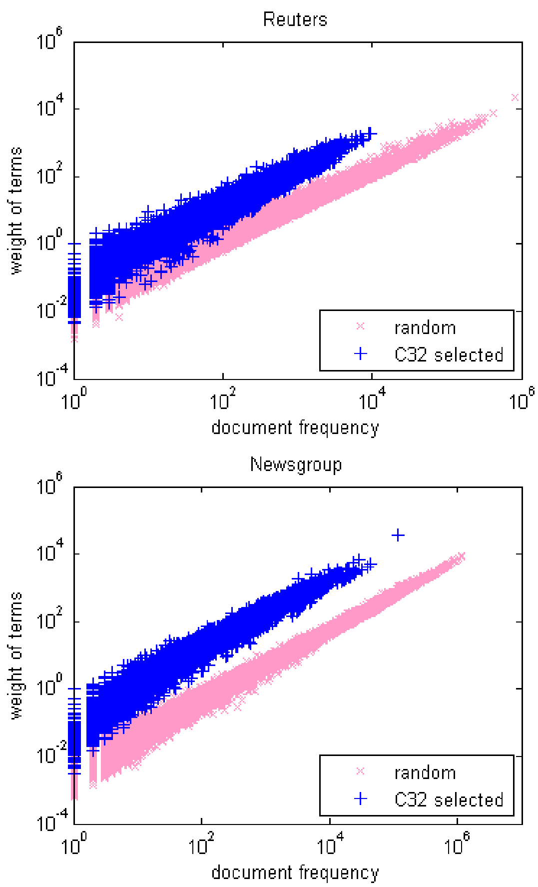

Figure 4.7: Weight of terms from a sample of *D* and weight of random words distribution over df. (Switched)

Table 4.12: Coverage of queries when query document frequencies are smaller then a certain value. Queries are from Webster dictionary. It shows that rarer words can not cover all the data source.

		$df < 100$	$df < 200$	$df < 400$	$df < 800$
English Wikipedia	queries	15225	18738	22206	25427
	coverage	260874	446237	689076	946498
Reuters	queries	11739	13540	15101	16434
	coverage	165263	261102	374189	493587
GOV2 subset0	queries	16608	19401	21816	24086
	coverage	154577	234059	322663	427560
Newsgroup	queries	12023	14660	17322	19935
	coverage	213705	386514	625135	890285
		$df < 1600$	$df < 3200$	$df < 6400$	$df < 12800$
English Wikipedia	queries	28133	30436	32219	33489
	coverage	1161432	1316559	1408958	1453897
Reuters	queries	17465	18322	18989	19458
	coverage	601011	691837	749629	786651
GOV2 subset0	queries	25914	27376	28556	29432
	coverage	531493	608433	680575	750637
Newsgroup	queries	22271	24339	25854	27037
	coverage	1105133	1239541	1295432	1368686
		$df < 25600$	$df < 51200$		
English Wikipedia	queries	34451	35069		
	coverage	1472563	1474847		
Reuters	queries	19787	20007		
	coverage	802648	806460		
GOV2 subset0	queries	30111	30615		
	coverage	828331	1030068		
Newsgroup	queries	27918	28580		
	coverage	1371701	1371958		

Table 4.13: The statistical data of query pools of C1, C2, C31 and C32.

Method		**Reuters**	**GOV2 subset0**	**Newsgroup**	**English Wikipedia**
C1	**max df**	806791	696878	1165272	1066195
	min df	0	0	0	0
	mean df	1283	4399	5770	4511
	df RSD	8.5555	5.5950	6.2986	5.5338
	max weight	22003.45	8724.58	8967.26	12202.55
	min weight	0	0	0	0
	W(QP,D)	20.1703	26.9230	34.3247	36.8789
	weight SD	221.3443	190.3310	237.2438	236.2848
	weight RSD	10.9738	7.0695	6.9118	6.4069
C2	**max df**	185570	474991	882129	691172
	min df	1	0	0	0
	mean df	217	879	911	827
	df RSD	12.9602	10.0205	13.6673	11.7910
	max weight	26486.88	20681.30	44009.47	48981.93
	min weight	0.0010	0	0	0
	W(QP,D)	19.9067	26.6720	34.0120	36.5499
	weight SD	281.9400	315.0866	520.7704	511.6770
	weight RSD	14.2345	11.8137	15.3114	13.9994
C31	**max df**	20676	82144	66243	31228
	min df	2	2	2	2
	mean df	951	147	929	517
	df RSD	1.3219	0.6440	2.5506	0.1358
	max weight	4339.4	6781.75	15171.15	3122.23
	min weight	0.02	0.001	0.001	0.001
	W(QP,D)	73.6374	14.4190	79.9570	37.5847
	weight SD	122.9127	189.2918	225.0074	70.1633
	weight RSD	1.6691	13.1279	2.8141	1.8668
C32	**max df**	24758	51432	119457	6821
	min df	1	1	1	1
	mean df	228	79	1042	77
	df RSD	2.9443	1.8635	2.3966	0.8477
	max weight	7799.26	16913.84	2325.06	2387.61
	min weight	0.01	0.001	0.0006	0.002
	W(QP,D)	36.9721	8.6577	14.6051	23.3278
	weight SD	139.6667	185.5397	33.4347	93.1826
	weight RSD	3.7776	21.4305	2.2892	3.9945

Chapter 5

The Comparison

This chapter compares the Coverage Method (C32) with several existing methods, including the CH-Reg method, Broder's method, the OR Method. Each method has its own restriction(s) to choose queries and process returned documents as stated in Chapter 2. In this chapter, we describe the experiments of the CH-Reg method, Broder's method and the OR Method.

5.1 The experiment of the CH-Reg Method

In our experiments, we use a query pool of 40,000 words randomly selected from the Webster Dictionary for the English corpora. In each experiment, the conditions are the same as those reported in [40], i.e., 5000 words are randomly selected from this query pool, ignoring the queries that match less than 20 documents, and take only the top 10 matched documents for estimation. The results are tabulated in Table 5.1.

We can draw a few conclusions from Table 5.1: i) Estimating the corpora which are larger than 500k has a larger bias than estimating smaller ones. ii) This method fails to

Table 5.1: A summary of the CH-Reg Method experiment on English corpora. Cells marked by '-' mean data are not available. Each trial randomly selects 5,000 queries the Webster Dictionary and discards any query that returns less than 20 documents. In each query, only top 10 matched documents are returned.

Reuters	N	100,000	500,000	806,791	-
	mean n	9,589	15,431	17,115	-
	RB	-0.3638	-0.5244	-0.5986	-
	RSD	0.0919	0.1060	0.1147	-
Newsgroup	N	100,000	500,000	1,372,911	-
	mean n	17,943	25,496	29,924	-
	RB	-0.0231	-0.2285	-0.4998	-
	RSD	0.0463	0.0695	0.0718	-
English Wikipedia	N	100,000	500,000	1,475,022	-
	mean n	19,528	29,581	35,835	-
	RB	0.1039	-0.0271	-0.3269	-
	RSD	0.0443	0.0518	0.0790	-
GOV2	N	100,000	500,000	1,077,019	2,000,000
	mean n	15,528	24,158	29,438	32,543
	RB	-0.4921	-0.5981	-0.7456	-0.8085
	RSD	0.0425	0.0556	0.0559	0.0628

work when estimating the size of GOV2. iii) Because 5,000 queries are issued in each trial, this method has a low variance, i.e., estimates on almost all the corpora have RB < 0.12, meaning this method has a low variance.

5.2 The experiments of Broder's method

In the experiments, we choose all 5-digit numbers as the first query pool (QP_A). Broder et al. constructed the second query pool (QP_B) by examining the corpus index directly. In our experiments, we repeat their way of constructing the second query pool, which consists of the medium frequency terms from each corpus index. Using this approach to constructing a query pool is impractical in the real-life application. We improve the construction using a similar approach to C32, i.e., issue random words from the Webster Dictionary, collect 3000 documents and extract terms from the sample to be QP_B. Two versions of Broder's method are denoted in Table 5.2.

Table 5.2: Broder's method notations

Notation	**D is transparent**
B0	$0 - No$
B1	$1 - Yes$

5.2.1 Constructing QP_B by terms from D

Broder's approach to constructing the second query pool (QP_B) requires to scan the corpus index and extract all terms and their *df* in the corpus. As the first query pool is all 5-digit numbers, in order to reduce the correlation of two query pools, when building the second query pool, we discard the terms containing any digits. After extraction, we sort the terms by their frequency. Beginning from 1/3 of the corpus size, we try to search a set of 100,000 consecutive terms such that these terms can capture about 30% of the corpus. In this way, we obtain the second query pool for each corpus. Table 5.3 records the coverage of each corpus. Tables 5.4 records the experimental data.

Table 5.3: The coverage of medium frequency terms of its corpus

Corpus	Reuters	English Wikipedia	Newsgroup	GOV2 subset0
Coverage	30.80%	36.48%	27.79%	30.45%

Table 5.4: Estimation using Broder's method - B1. RB and RSD are calculated by 100 trials.

Corpus	Metric	**t = 5-digit numbers + medium frequency terms**				
		200	1000	2000	4000	5000
Reuters	mean n	655	3,207	6,406	12,801	16,093
	RB	0.6559	0.2778	0.2665	0.2424	0.2659
	RSD	0.7608	0.2637	0.1868	0.1446	0.1661
English Wikipedia	mean n	1,343	6,788	13,560	27,019	33,800
	RB	-0.3737	-0.3849	-0.3833	-0.3842	-0.3831
	RSD	0.0993	0.0145	0.0113	0.0080	0.0066
Newsgroup	mean n	1,004	5,075	10,178	20,265	5,067
	RB	-0.1349	-0.2844	-0.3373	-0.2861	-0.2281
	RSD	1.2387	0.6320	0.4176	0.3536	0.7169
GOV2 Subset0	mean n	2,365	12,818	27,258	51,704	62,901
	RB	-0.5010	0.0450	1.4174	-0.1302	-0.3242
	RSD	1.6313	4.0101	5.5971	1.6627	1.3089

The reason why it uses a set of medium frequency terms from the corpus but not random words is that: A few high frequency words could capture a set of documents that have a high coverage of the corpus, often as high as 95% as shown in Table 4.3, which makes $|M(QP_B, D)|$ very close to N. The other drawback is that only a few words have high weight of terms. When randomly selecting words from this kind of query pool, if those high-weight terms are selected, it would bring a large variance when estimating $|M(QP_B, D)|$. This is proven by the data of C1 in Table 4.3. We tried to use 40,000 random words as QP_B

from Webster dictionary. Even randomly selecting 500 queries from each query pool, the $|M(QP_A, D) \hat{\cap} M(QP_B, D)|$ is always equals to $|M(Q\hat{P}_A, D)|$. This means that $M(QP_B, D)$ has already included $M(QP_A, D)$, making QP_A useless.

5.2.2 Constructing QP_B by terms from a sample of D

In this section, we use a query pool learnt from a sample of D instead of considering D is transparent. The reason is that in the real application, usually a corpus or a data collection is considered as a black box. It is impossible to obtain the df of terms in advance. As we discussed in Section 4.4, learning queries from a sample of D has to configure three parameters:

- In order to make it comparable with C32, we chose a sample size at 3,000.
- For the starting df, we also set it to 1 to maintain a low variance.
- Because $M(QP_B, D)$ can not be too large as stated in the last section, we set the coverage to 40%.

The experimental data are recorded in Table 5.5.

5.2.3 Summary

We summarize B0 and B1 in this section. As we can observe from Table 5.4 and Table 5.5, B0 needs to sample more documents. This is because QP_B are not low frequency words although they have low df in the sample of D. Figure 5.1 shows a comparison between B1 and B0 in MSE-Cost plots.

Figure 5.1 demonstrates B1 can estimate the size of four English corpora more accurately than B0. The cost is less as well, even while issuing 5000 queries. Estimating the

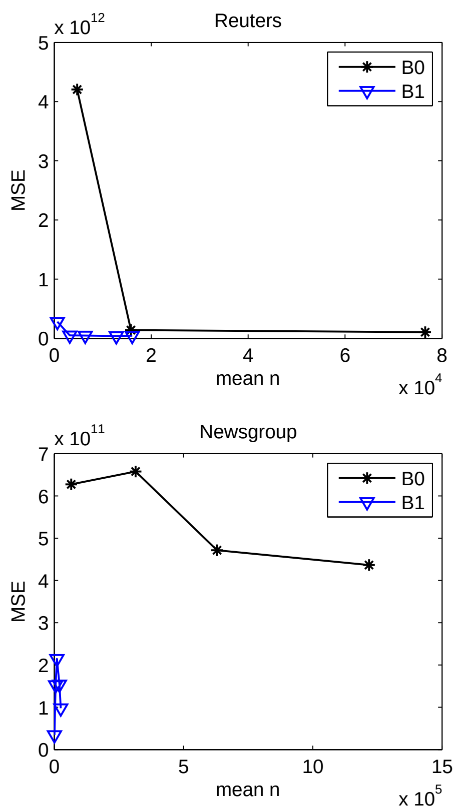

Figure 5.1: A comparison of Broder' method - B1 and B0. Data are collected from Table 5.4 and Table 5.5.

Table 5.5: Estimation using Broder's method - B0. RB and RSD are calculated by 100 trials.

Corpus	Metric	**t = 5-digit numbers + medium frequency terms**				
		50	100	200	1000	2000
Reuters	mean n	4,724	15,815	76,514	151,000	301,400
	RB	0.1653	-0.3531	-0.3916	-0.3996	-0.4032
	RSD	2.1759	0.4642	0.1295	0.0710	0.0591
GOV2 subset0	mean n	4,539	8,781	14,866	81,472	161,498
	RB	-0.3772	-0.4439	-0.5401	-0.4340	-0.1726
	RSD	1.3691	1.4593	0.5714	0.5138	2.1090
Newsgroup	mean n	16,508	31,144	65,475	314,656	629,839
	RB	6.5787	0.0563	-0.2526	-0.4018	-0.4408
	RSD	7.3737	1.8539	0.6939	0.7236	0.4223
English Wikipedia	mean n	3,260	5,896	11,644	59,430	121,158
	RB	-0.4504	-0.5174	-0.5213	-0.5123	-0.5049
	RSD	0.4589	0.3176	0.2431	0.1138	0.0805
GOV2 2M	mean n	21908	45,497	92,976	454,214	920,182
	RB	-0.5990	-0.4803	-0.5510	-0.5669	-0.4853
	RSD	0.7938	0.9744	0.5278	0.3433	1.5719

size of the GOV2 subset0 seems difficult because the points are spread in a wide range on the y-axis. This is because the variance of the estimates is large. We will discuss why the estimated size of the GOV2 subsets are varied so much in the later sections.

The reason we are not able to obtain the data of Broder's method on small collections is that the estimate of $|M(QP_A, D)|$ has a high chance to be 0.

5.3 The experiment of the OR Method

We also carry out experiments to collect data on the OR Method. The query pool used for estimating the size of English corpora is 40,000 Webster words. In each trial, the top 2% of queries are removed. Data are presented in Table 5.6 and Table 5.7.

Table 5.6: Estimating small English corpora using the OR Method. Bias and standard deviation of the estimation over 100 trials. In each trial, queries are randomly selected from 40,000 Webster words.

Corpus	Metric	t				
		50	100	200	500	1000
Reuters 100k	mean n	2,750	5,072	9,342	20,421	40,742
	RB	859.4513	0.1157	-0.0033	-0.0363	-0.0595
	RSD	4.9150	0.3202	0.1818	0.0973	0.0681
Reuters 500k	mean n	12,729	23,636	43,406	104,876	210,658
	RB	0.4713	0.1249	0.0004	-0.0489	-0.0489
	RSD	0.6580	0.3209	0.1822	0.1065	0.0652
GOV2 100k	mean n	8,849	15,744	33,017	72,943	139,084
	RB	-0.0521	-0.2193	-0.2444	-0.1627	-0.1328
	RSD	0.7530	0.4112	0.3017	0.1854	0.1440
GOV2 500k	mean n	51,786	90,820	147,408	360,404	756,560
	RB	-0.0227	-0.1454	-0.2679	-0.2298	-0.1178
	RSD	0.6258	0.5203	0.3508	0.1988	0.1219
Newsgroup 100k	mean n	1,898	3,491	7,150	17,686	35,449
	RB	0.0437	-0.2009	-0.2161	-0.1653	-0.0996
	RSD	0.6158	0.2261	0.1304	0.0576	0.0280
Newsgroup 500k	mean n	52,116	78,209	149,500	403,871	779,771
	RB	0.5835	0.1106	0.0577	0.0976	0.1301
	RSD	0.9479	0.2736	0.1394	0.0600	0.0294
English Wikipedia 100k	mean n	7,489	14,052	25,269	60,406	123,635
	RB	-0.1326	-0.1581	-0.1922	-0.1300	-0.0593
	RSD	0.2647	0.1942	0.1296	0.0768	0.0395
English Wikipedia 500k	mean n	34,809	68,943	127,146	322,304	642,041
	RB	-0.1089	-0.1927	-0.1942	-0.1276	-0.0451
	RSD	0.3903	0.1896	0.1078	0.0700	0.0406

Table 5.7: Estimating large English corpora using the OR Method. Bias and standard deviation of the estimation over 100 trials. In each trial, queries are randomly selected from 40,000 Webster words.

Corpus	Metric	t				
		50	100	200	500	1000
Reuters	mean n	29,155	43,455	70,942	169,767	329,290
	RB	27.3230	0.1380	0.0249	-0.0579	-0.0615
	RSD	9.3919	0.2861	0.1819	0.1038	0.0673
GOV2 subset0	mean n	95,060	180,398	367,925	783,734	1,642,266
	RB	-0.0696	-0.1710	-0.2291	-0.1980	-0.0856
	RSD	0.7802	0.4863	0.2670	0.1846	0.1312
Newsgroup	mean n	137,403	242,043	438,959	1,047,621	2,110,965
	RB	0.2709	0.1349	0.0541	0.0945	0.1250
	RSD	0.3941	0.2275	0.1337	0.0606	0.0254
English Wikipedia	mean n	114,837	202,994	371,907	969,385	1,864,510
	RB	-0.0571	-0.1568	-0.1974	-0.1153	-0.0598
	RSD	0.3799	0.1758	0.1116	0.0679	0.0462
GOV2 2M	mean n	163,363	304,809	595,761	1,451,201	3,063,481
	RB	0.1283	-0.3187	-0.2919	-0.2253	-0.0975
	RSD	1.2200	0.5443	0.3341	0.1946	0.1334

5.4 The experiments on Chinese corpora

We experiment the CH-Reg method, the OR method and C32 on Chinese corpora. B0 and B1 are not easy to implement because it is difficult to choose two uncorrelated query pools for Chinese corpora. Take the Sogou Web Corpus 500k for example, we tried to query all 5-digit numbers on this collection, but no document matched them.

When carrying out the experiments on Chinese corpora, we need to decide whether the queries should be single Chinese characters or Chinese phrases for the CH-Reg Method and OR Method. In general, Chinese characters have much higher dfs than English words because the number of characters are usually limited to be a few thousands in most corpora. Thus a few hundreds of random Chinese characters may match most of the documents. On the other hand, using phrases will reduce the cost because phrases have lower df than characters. The dictionary we used is the Contemporary Chinese Dictionary, which contains 44,905 phrases. Queries selected by the CH-Reg Method and OR Method are phrases.

When creating a sample of a Chinese collection using C32, we use phrases as queries. After documents are collected by queries, it is difficult to extract all the phrases from a Chinese document because the Chinese segmentation is still a problem under research. Hence, the terms collected from a sample of D are all the terms tokenized by Lucene. By inspecting the query pools built by C32 from Chinese documents, we found that these terms consist of not only Chinese characters but also some other symbols in the documents.

The data of the CH-Reg Method, the OR Method and C32 are tabulated in Table 5.8, Table 5.9 and Table 5.10.

Table 5.8: A summary of the CH-Reg Method experimental data on Chinese corpora. Cells marked by '-' mean data are not available. Each trial randomly selects 5,000 queries from the Contemporary Chinese Dictionary and discards any query that returns less than 20 documents. In each query, only top 10 matched documents are returned. Bias and standard deviation of the estimation over 100 trials.

Chinese Wikipedia	N	212,042	-	-	-
	mean n	29,978	-	-	-
	RB	0.1888	-	-	-
	RSD	0.0363	-	-	-
Chinese Literature	N	90,749	-	-	-
	mean n	33,573	-	-	-
	RB	-0.0696	-	-	-
	RSD	0.0230	-	-	-
Sogou Web Corpus	N	100,000	500,000	1,000,000	2,000,000
	mean n	26,247	37,155	40,657	43,581
	RB	0.2403	0.1066	-0.5986	-0.7331
	RSD	0.0271	0.0463	0.0446	0.0435

Table 5.9: Estimating Chinese corpora using the OR Method. Bias and standard deviation of the estimation over 100 trials. In each trial, queries are randomly selected from Contemporary Chinese Dictionary.

Corpus	Metric			t		
		50	100	200	500	1000
Chinese Literature	mean n	11,725	23,642	44,073	111,761	224,151
	RB	-0.5318	-0.5406	-0.5444	-0.5292	-0.4903
	RSD	0.1629	0.1071	0.0611	0.0418	0.0355
Chinese Wikipedia	mean n	12,035	22,486	41,268	105,735	208,582
	RB	-0.3971	-0.4405	-0.4263	-0.3602	-0.2838
	RSD	0.3332	0.1584	0.1096	0.0640	0.0397
Sogou Web Corpus	mean n	40,507	73,214	140,643	345,499	687,370
500k	RB	-0.3272	-0.3606	-0.3604	-0.2888	-0.2220
	RSD	0.3088	0.2027	0.1203	0.0643	0.0473
Sogou Web Corpus	mean n	78,800	142,326	283,121	693,748	1,344,885
1M	RB	-0.2840	-0.3600	-0.3570	-0.2935	-0.2332
	RSD	0.3976	0.1696	0.1146	0.0605	0.0431

Table 5.10: The estimation by C32 on Chinese corpora. Data are obtained by 100 trials, each trial is produced by randomly selecting t number of queries from QP.

Corpus	Metric	t				
		50	100	200	500	1000
Chinese Literature	mean n	7,386	14,634	30,003	70,683	143,548
	RB	-0.0404	-0.0751	-0.0471	-0.0995	-0.0816
	RSD	0.4768	0.2797	0.2370	0.1675	0.0991
Chinese Wikipedia	mean n	1,778	3,699	7,613	18,157	36,165
	RB	-0.1495	-0.1005	-0.0525	-0.1026	-0.1062
	RSD	0.5466	0.3730	0.3268	0.2536	0.1571
Sogou Web Corpus 500k	mean n	66,401	128,770	256,442	653,664	1,306,888
	RB	-0.1260	-0.1541	-0.1557	-0.1311	-0.1303
	RSD	0.3389	0.2182	0.1732	0.1008	0.0676
Sogou Web Corpus 1M	mean n	98,943	197,521	390,273	975,726	1,943,283
	RB	-0.1537	-0.1554	-0.1772	-0.1763	-0.1824
	RSD	0.3253	0.2232	0.1700	0.0956	0.0693

Table 5.11: The size and coverage of QP for each corpora in Table 5.10.

Corpus	Chinese Literature	Chinese Wikipedia	Sogou Web Corpus 500k	Sogou Web Corpus 1M
$\|QP\|$	14,484	95,238	4,464	4,547
Coverage in D	93.10%	92.33%	86.89%	82.19%

5.5 The comparative study

In this section, we visualize the data obtained in the above experiments using MSE-Cost and MSE-Queries plots. As the MSE is a combination of variance and bias, it is obvious that the smaller MSE the better, and so does to the cost. Generally, any point that is closer to (0,0) in the two dimensional space means the method it belongs to can perform better. However, those points that indicate a low MSE but a slightly higher cost should also be considered acceptable. From the tables in previous sections, we can have many combinations of MSE and cost. In the plots, we remove any point that has an extremely large MSE or 'mean n' which results in other points gathering in a narrow area. As mentioned in Chapter 2, there are basically two approaches to estimation. B0 and C32 need to download and exam document content. We compare them in MSE-Cost plots, |RB|-Cost plots and RSD-Cost plots. The other approach only checks IDs, and its cost does not include document downloading. We still provide an overall picture of the C32 with the CH-Reg Method and the OR Method in MSE-Cost plots of small English corpora and Chinese corpora.

5.5.1 Methods that need to download documents

In this section, we compare C32 with its direct competitor - Broder's method practical version (B0). The updated Broder's method (B0) does not require to know the terms with *df* in advance. Therefore, it is comparable with C32.

Figure 5.2 plots the practical version of Broder's method (B0). From this figure, we can easily draw the conclusion that taking the same size of sampled documents from D, C32 works much better in the corpora other than GOV2 subset0. B1 works best for Newsgroup. Even when estimating the size of the GOV2 subset0, C32 is able to achieve similar performance to B0.

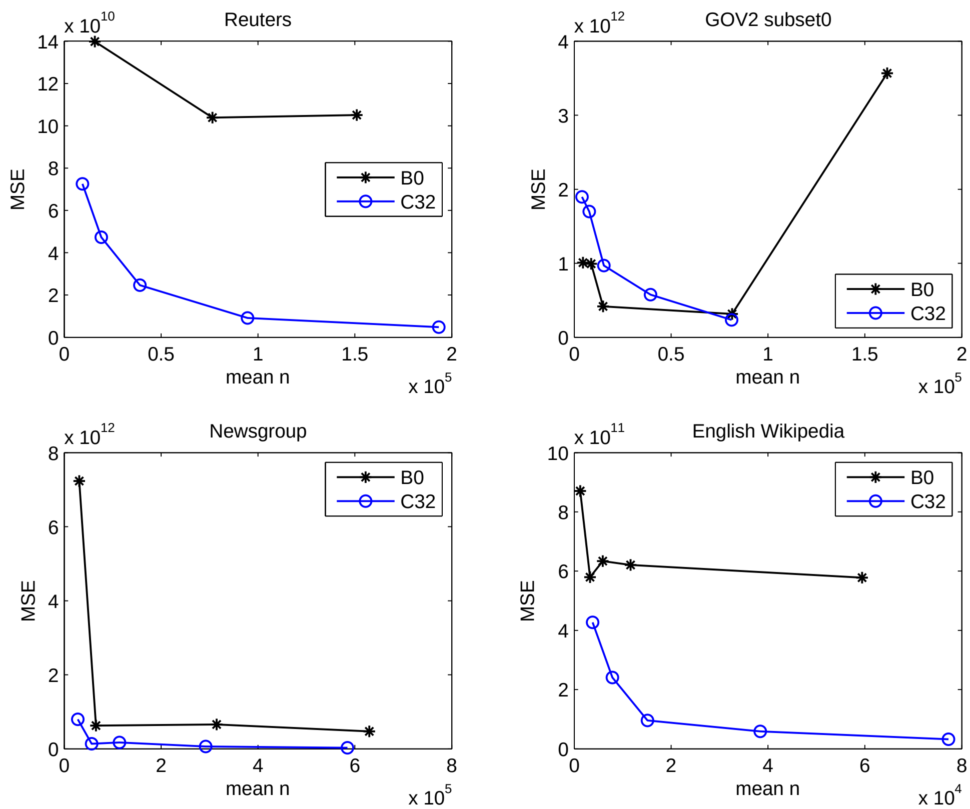

Figure 5.2: A comparison of C32 and Broder's Method practical version (B0) on large English Corpora by MSE-Cost plots. Data are obtained from Table 5.5 and Table 4.10.

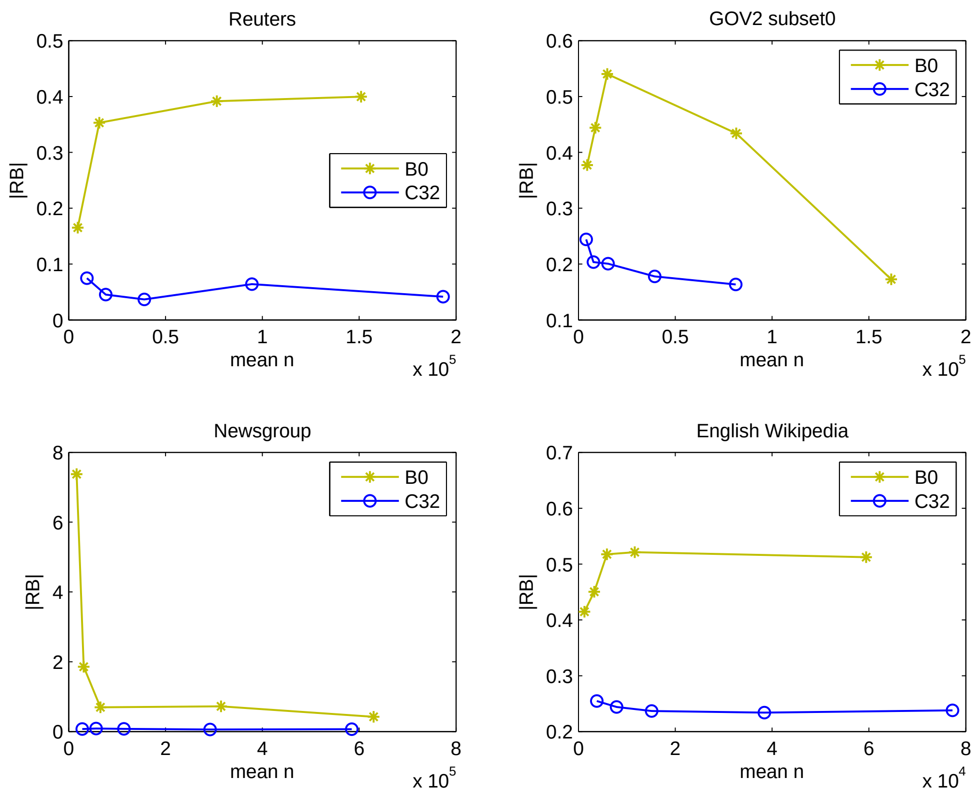

Figure 5.3: A comparison of C32 and Broder's Method practical version (B0) on large English Corpora by RB-Cost plots. Data are obtained from Table 5.5 and Table 4.10.

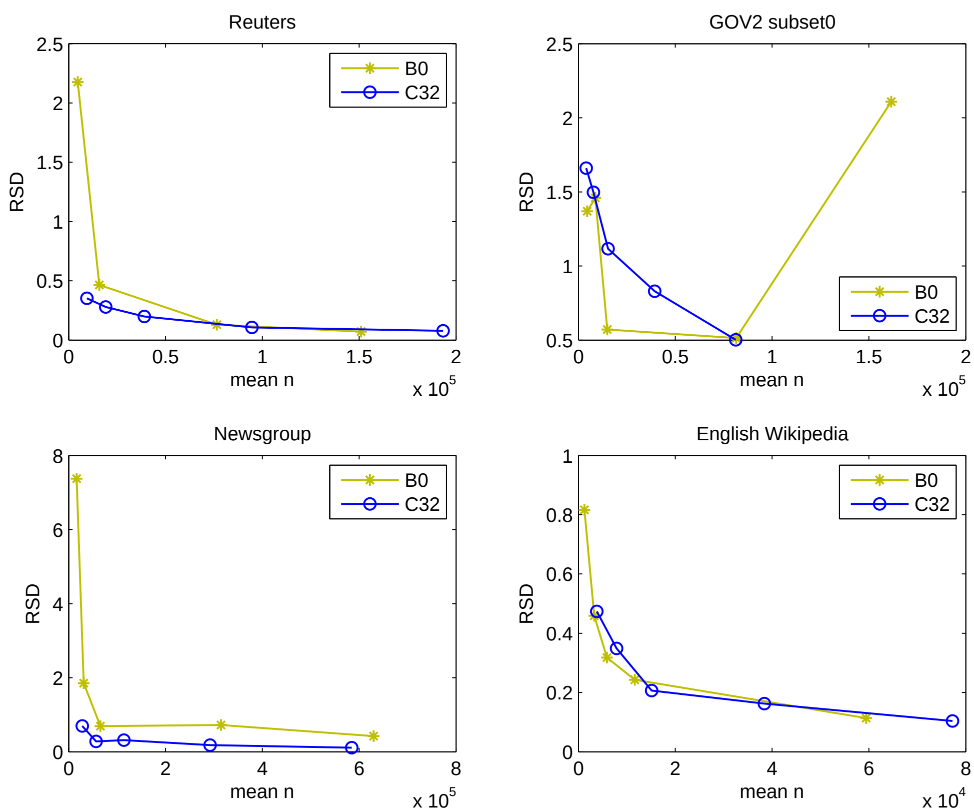

Figure 5.4: A comparison of C32 and Broder's Method practical version (B0) on large English Corpora by RSD-Cost plots. Data are obtained from Table 5.5 and Table 4.10.

5.5.2 Methods that only need to check IDs

In this section, we provide visualized data that show the performance of the OR Method and the CH-Reg Method. Figure 5.5 and Figure 5.6 demonstrate the CH-Reg method and the OR method when estimating small English corpora. Figure 5.7 presents the result for the Chinese corpora. The plots also included the data of C32. We intend to show how well our proposed method can do when compared with the CH-Reg Method and the OR Method does.

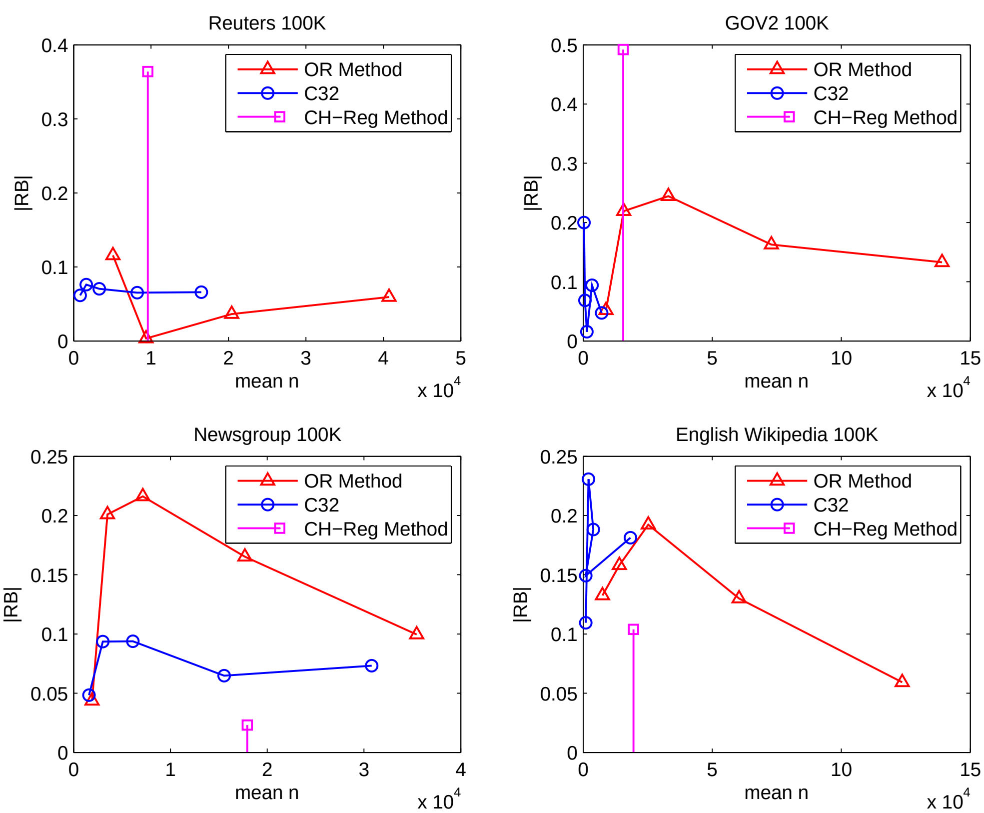

Figure 5.5: The OR Method, CH-Reg Method and C32 on 100,000 documents English Corpora. Data are obtained from Table 5.1, Table 5.6 and Table 4.11.

Figure 5.5 indicates that C32 can achieve higher accuracy than the CH-Reg method for estimating Reuters and GOV2. The CH-Reg Method works best for 100k Newsgroup and English Wikipedia.

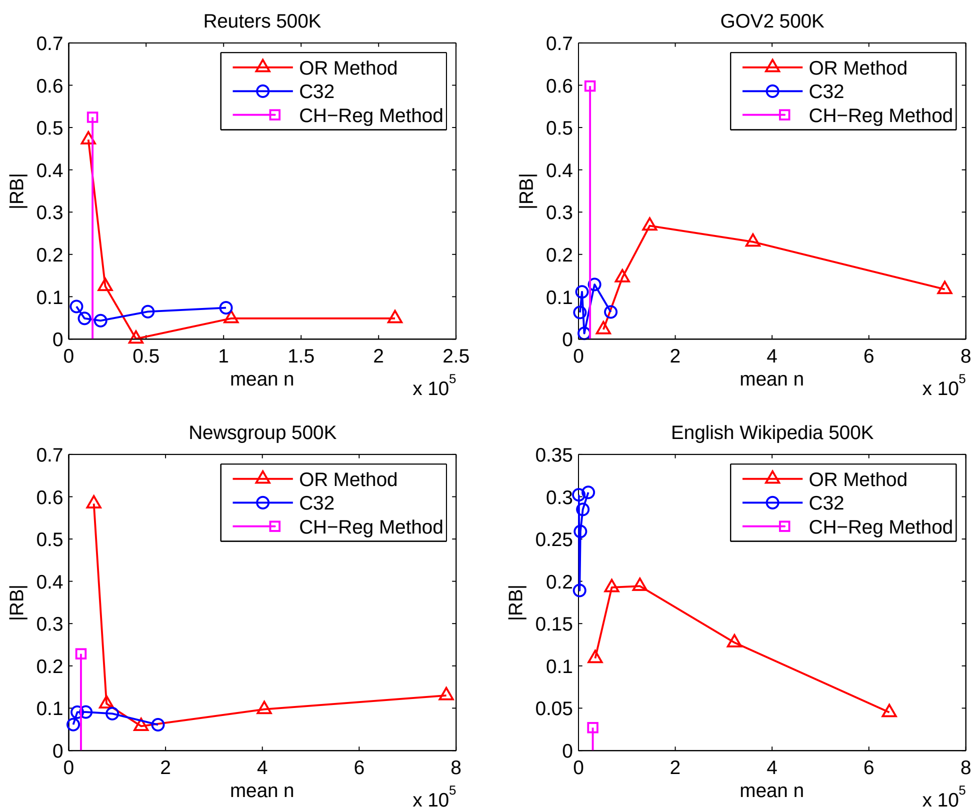

Figure 5.6: The OR Method, CH-Reg Method and C32 on 500,000 documents English Corpora. Data are obtained from Table 5.1, Table 5.6 and Table 4.11.

Figure 5.6 shows that when estimating the collections that have 500k documents, C32 and the OR Method can be much more accurate than the CH-Reg except for English Wikipedia.

From Figure 5.7 we can see that the CH-Reg Method and C32 work better for the small Chinese corpora. C32 and the OR Method work best for large Chinese corpus. The best

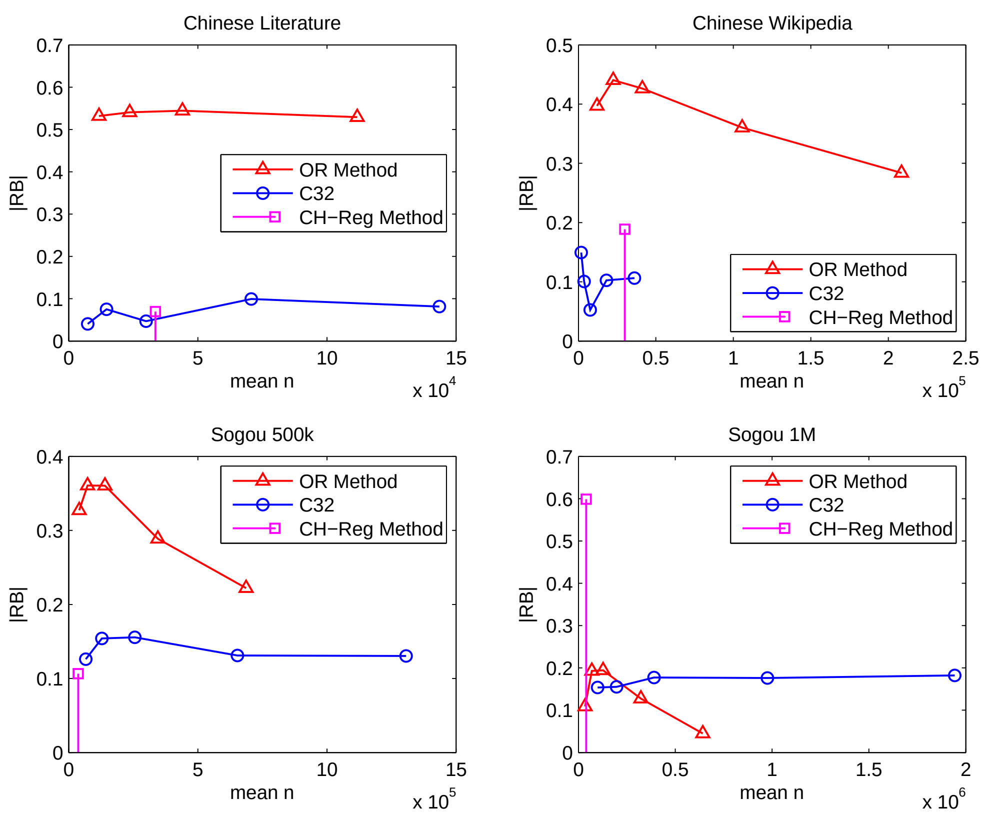

Figure 5.7: The OR Method, C32 and CH-Reg Method on Chinese Corpora. Data are obtained from Table 5.8, Table 5.9 and Table 5.10.

estimate C32 can produce is always better than that of the OR Method.

5.5.3 The overall comparison

Figure 5.8 and Figure 5.9 show the performance of four methods on four large English corpora. Note that in Figure 5.9, the data of B1 is obtained from Table 5.4 which is impractical due to the assumption of the transparency of the corpora.

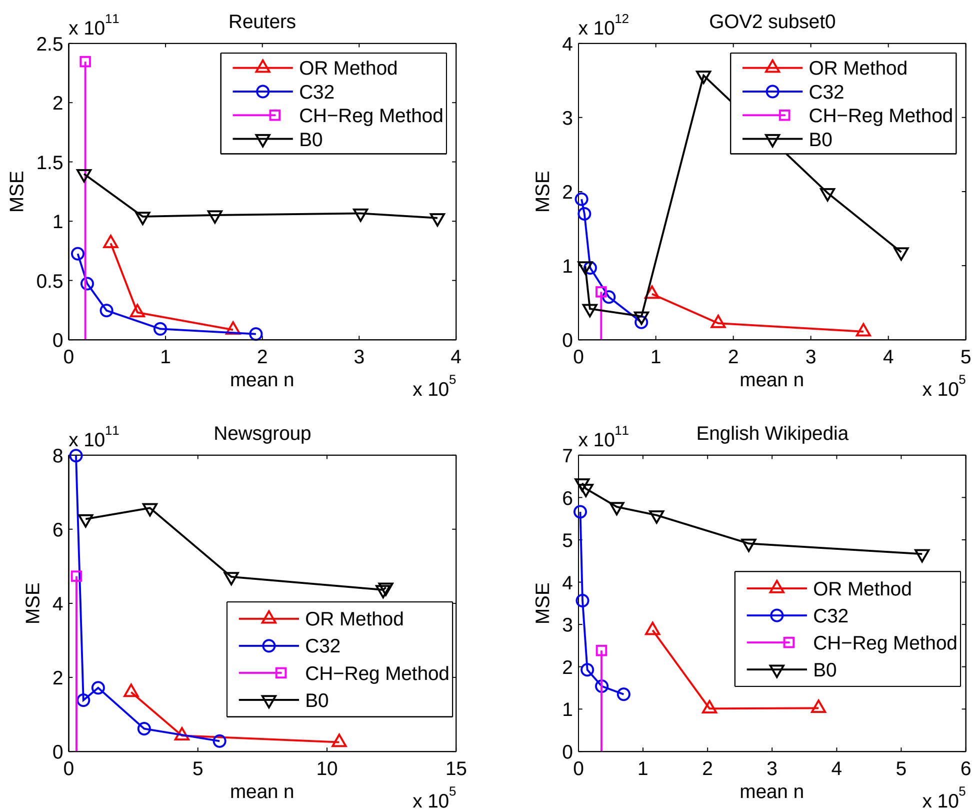

Figure 5.8: The OR Method, C32, CH-Reg Method and B0 on English Corpora. Data are obtained from Table 5.1, Table 5.5, Table 5.7 and Table 4.10.

As we can see from Figure 5.8, C32 and the OR Method are able to produce relative

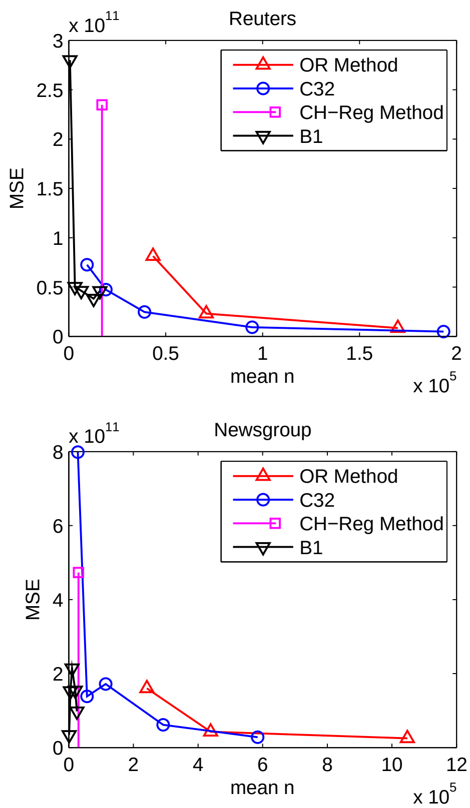

Figure 5.9: The OR Method, C32, CH-Reg Method and B1 on English Corpora. Data are obtained from Table 5.1, Table 5.4, Table 5.7 and Table 4.10.

low MSE to estimate the size of Reuters, while C32 and B1 have similar cost but produce lowest MSE. None of the methods work well for estimating GOV2's size. But C32, B0 and the CH-Reg method can estimate its size more accurately by checking around 2,500 documents. C32 works better than the OR method for Newsgroup and English Wikipedia. Moreover, if we need low variance and high accuracy to estimate a size, only C32 and OR method can be applied, as checking more documents will lead to a very low MSE.

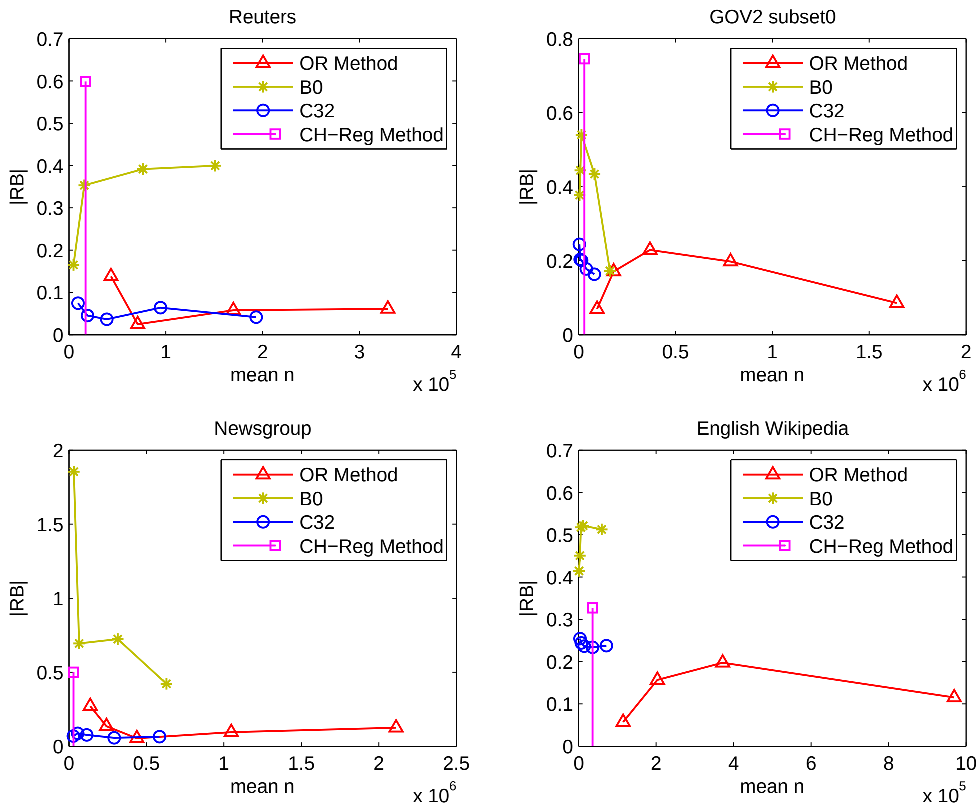

Figure 5.10: The OR Method, C32, CH-Reg Method and B0 on English Corpora. Data are obtained from Table 5.1, Table 5.4, Table 5.7 and Table 4.10.

From Table 5.4, Table 5.5, Figure 5.8, Figure 5.9 and Figure 5.2 we found that estimating the size of GOV2 has a large variance, and hence produces varied bias. The first reason is that from a sample of GOV2, terms are of a large range of weights. We know from Table 4.3 that random words result in a high variance of estimation. Figure 4.6 illustrates the weight-df distribution of random words and terms selected by C32 in four corpora. The selected terms also show similar weight-df distribution of random words. Moreover, the second reason is that the variance of weight of terms selected from a sample is large. In Table 4.13 we can see that the RSDs of weight of terms selected by C31 from a sample of Reuters, Newsgroup or English Wikipedia are less than 4; much lower than that from a sample of GOV2. This indicates estimating the size of GOV2 would have larger variance.

Chapter 6

Conclusions

This thesis proposes a method to estimate the size of a deep web data source. This method relies on the identification of a set of queries that can match almost all the documents in a data source. In addition, these queries should have similar document frequencies so that the variance is small. We examine two approaches of constructing a query pool. Our experiments in the four data collections show that the queries learnt from a sample can produce low variance, high accuracy and low cost. Our method usually underestimates N. This feature is best for deep web crawling algorithms because any overestimate of N will cause a non-stop crawling process. The three parameters in our method can be changed according to requirements in different circumstances.

This thesis also compares the new method with three existing methods in terms of variance, bias and cost. We compared our method with the method proposed by Broder et al. which are also need to download and analyze documents. We conclude that the proposed method (C32) is better than Broder's method (B0) which considers the data source is not transparent. However, when comparing our method with the methods that only analyze document IDs, it is hard to draw a conclusion about which method works the best for any size of collections in Chinese and English. However, the OR Method and Pool-base Coverage Method are capable to produce small variance and bias when issuing more queries, although this makes the cost more expensive. The CH-Reg Method has very small variance. However, it produces a large bias when estimating most of the collections. We summarize the advantages of each method in Table 6.1. The *Queries* provides the numbers of queries a method needs to achieve low variance and high accuracy. The *Stability* measures whether a method can produce similar bias and variance when estimating all the data collections. The *Cost*, *Bias* and *Variance* roughly show a method's performance among all the methods.

Table 6.1: A summary of all methods. The measurement tagged by '/e' mean it has exception(s).

Metric	Cost	Queries	Bias	Variance	Stability	Download
CH-Reg Method	low	5000	varied	small	low	No
OR Method	low	100-200	medium	small	good	No
B1	low	5000	large	small/e	good	Yes
B0	low	2000	large	large	good	Yes
C1	high	500	small	medium	moderate	Yes
C2	medium	500	small	large	good	Yes
C31	low	200-500	small	small/e	moderate	Yes
C32	low	1000	small/e	small/e	moderate	Yes

Our method has a few limitations. One is that the method needs to download the documents, which is not only costly but also sometimes impossible. The second limitation is that our method needs to download ALL the documents that match the queries. Many data sources return only top-k matched documents. This results in the calculating error of the query weights. Therefore, the estimator becomes biased. The third one is caused by the coverage of a sample of the data source. A query pool has a high coverage in a sample cannot always has a similar coverage in the data source. The fourth problem is the variance of this method. According to the $weight - df$ distribution of term weights, in order to i) make the query pool reasonably large, ii) choose queries that have similar dfs, iii) choose terms that can match as less documents as possible, we need to collect the terms with low dfs. In a sample of the data source, the selected terms have dfs equal to 1 or 2. But theses queries have a much wider range of (1,2) in the data source. It implies that the variance of weights in the data source is difficult to predict from its sample.

Bibliography

[1] Gustavo Alonso, Fabio Casati, Harumi Kuno, and Vijay Machiraju. *Web Services*. Springer, 1st edition, October 2003.

[2] Steven C. Amstrup, Trent L. McDonald, and Bryan F. J. Manly. *Handbook of Capture-Recapture Analysis*. Princeton University Press, October 2005.

[3] Apache.org. Apache lucene. http://lucene.apache.org/java/docs, December 2007.

[4] Edward Bruce Banning. *The Archaeologist's Laboratory: The Analysis of Archaeological Data*. Springer, may 2005.

[5] Ziv Bar-Yossef and Maxim Gurevich. Random sampling from a search engine's index. In *Proceedings of the 15th international conference on World Wide Web*, pages 367–376, Edinburgh, Scotland, 2006. ACM.

[6] Ziv Bar-Yossef and Maxim Gurevich. Efficient search engine measurements. In *Proceedings of the 16th international conference on World Wide Web*, pages 401–410, Banff, Alberta, Canada, 2007. ACM.

[7] Luciano Barbosa and Juliana Freire. Siphoning hidden-web data through keyword-based interfaces. *In SBBD*, pages 309—321, 2004.

[8] Michael K Bergman. White paper - the deep web: Surfacing hidden value. *Journal of Electronic Publishing*, 7(1), 2001.

[9] Krishna Bharat and Andrei Broder. A technique for measuring the relative size and overlap of public web search engines. *Comput. Netw. ISDN Syst.*, 30(1-7):379–388, 1998.

[10] Allan Bluman. *Elementary Statistics: A Brief Version*. McGraw-Hill Science/Engineering/Math, 4th edition, December 2006.

[11] Igor Bolshakov and Sofia Galicia-Haro. *Can We Correctly Estimate the Total Number of Pages in Google for a Specific Language?*, volume 2588/-1 of *Lecture Notes in Computer Science*, pages 415–419. Springer Berlin / Heidelberg, 2008.

[12] Andrei Broder, Marcus Fontura, Vanja Josifovski, Ravi Kumar, Rajeev Motwani, Shubha Nabar, Rina Panigrahy, Andrew Tomkins, and Ying Xu. Estimating corpus size via queries. In *Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 594–603, Arlington, Virginia, USA, 2006. ACM.

[13] Rui Cai, Jiang-Ming Yang, Wei Lai, Yida Wang, and Lei Zhang. iRobot: an intelligent crawler for web forums. In *Proceeding of the 17th international conference on World Wide Web*, pages 447–456, Beijing, China, 2008. ACM.

[14] Jamie Callan and Margaret Connell. Query-based sampling of text databases. *ACM Trans. Inf. Syst.*, 19(2):97–130, 2001.

[15] Anne Chao. Estimating the population size for capture-recapture data with unequal catchability. *Biometrics*, 43(4):783–791, December 1987.

[16] Charlie Clarke and Ian Soboroff. Trec gov2 test collection. http://ir.dcs.gla.ac.uk/test_collections/gov2-summary.htm, July 2008.

[17] J. N. DARROCH. THE MULTIPLE-RECAPTURE CENSUS: i. ESTIMATION OF a CLOSED POPULATION. *Biometrika*, 45(3-4):343–359, December 1958.

[18] Ludovic Denoyer and Patrick Gallinari. The wikipedia xml corpus. *SIGIR Forum*, 40(1):64–69, 2006.

[19] A. Gulli and A. Signorini. The indexable web is more than 11.5 billion pages. In *Special interest tracks and posters of the 14th international conference on World Wide Web*, pages 902–903, Chiba, Japan, 2005. ACM.

[20] Panagiotis G. Ipeirotis and Luis Gravano. Distributed search over the hidden web: hierarchical database sampling and selection. In *Proceedings of the 28th international conference on Very Large Data Bases*, pages 394–405, Hong Kong, China, 2002. VLDB Endowment.

[21] Nobuko KISHI, Takahiro OHMORI, Seiji SASAZUKA, Akiko KONDO, Masahiro MIZUTANI, and Takahide OGAWA. Estimating web properties by using search engines and random crawlers. In *Proceeding of the Annual Internet Society Conference*, volume 2, Yokohama, Japan, July 2000. Internet Society.

[22] Sogou Lab. Sogou web corpus. http://www.sogou.com/labs/dl/t.html, December 2008.

[23] Steve Lawrence and C. Lee Giles. Searching the world wide web. *Science*, 280(5360):98–100, April 1998.

[24] Steve Lawrence and C. Lee Giles. Accessibility of information on the web. *Intelligence*, 11(1):32–39, 2000.

[25] Shen-Ming Lee and Anne Chao. Estimating population size via sample coverage for closed capture-recapture models. *Biometrics*, 50(1):88–97, March 1994.

[26] Jie Liang, Yanyin Zhang, Jianguo Lu, Sabiha Sathulla, Ding Chen, and Shaohua Wang. A rental advising system based on service oriented architecture. In *Proceedings of the 2008 IEEE Congress on Services - Part I*, pages 184–190. IEEE Computer Society, 2008.

[27] Stephen Liddle, David Embley, Del Scott, and Sai Yau. *Extracting Data behind Web Forms*, pages 402–413. Springer, 2003.

[28] King-Lup Liu, Clement Yu, Clement Yu, and Weiyi Meng. Discovering the representative of a search engine. In *Proceedings of the eleventh international conference on Information and knowledge management*, pages 652–654, McLean, Virginia, USA, 2002. ACM.

[29] Jianguo Lu. Efficient estimation of the size of text deep web data source. In *Proceeding of the 17th ACM conference on Information and knowledge management*, pages 1485–1486, Napa Valley, California, USA, 2008. ACM.

[30] Jianguo Lu and Dingding Li. Estimating deep web data source size by capture-recapture method. accepted. *Journal of Information Retrieval*, 2009.

[31] Jianguo Lu, Yan Wang, Jie Liang, Jessica Chen, and Jiming Liu. An approach to deep web crawling by sampling. In *2008 IEEE/WIC/ACM International Conference*

on Web Intelligence and Intelligent Agent Technology, volume 1, pages 718–724, Los Alamitos, CA, USA, 2008. IEEE Computer Society.

[32] 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, 2008.

[33] Rajeev Motwani, Rina Panigrahy, and Ying Xu. *Estimating Sum by Weighted Sampling*, volume 4596 of *Lecture Notes in Computer Science*, pages 53–64. Springer, 2007.

[34] Michael L. Nelson, Joan A. Smith, and Ignacio Garcia del Campo. Efficient, automatic web resource harvesting. In *Proceedings of the 8th annual ACM international workshop on Web information and data management*, pages 43–50, Arlington, Virginia, USA, 2006. ACM.

[35] Alexandros Ntoulas, Petros Zerfos, and Junghoo Cho. Downloading textual hidden web content through keyword queries. In *Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries*, pages 100–109, Denver, CO, USA, 2005. ACM.

[36] Sriram Raghavan and Hector Garcia-Molina. Crawling the hidden web. In *Proceedings of the 27th International Conference on Very Large Data Bases*, pages 129–138, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[37] Thomson Reuters. Reuters coprus. http://about.reuters.com/researchandstandards/corpus/, December 2008.

[38] F. X. Schumacher and R. W. Eschmeyer. The estimation of fish populations in lakes and ponds. *Journal of the Tennessee Academy of Science*, 18:228, 1943.

[39] Denis Shestakov, Sourav S. Bhowmick, and Ee-Peng Lim. DEQUE: querying the deep web. *Data & Knowledge Engineering*, 52(3):273–311, 2005.

[40] Milad Shokouhi, Justin Zobel, Falk Scholer, and S. M. M. Tahaghoghi. Capturing collection size for distributed non-cooperative retrieval. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 316–323, Seattle, Washington, USA, 2006. ACM.

[41] Luo Si and Jamie Callan. Relevant document distribution estimation method for resource selection. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 298–305, Toronto, Canada, 2003. ACM.

[42] Luo Si, Rong Jin, Jamie Callan, and Paul Ogilvie. A language modeling framework for resource selection and results merging. In *Proceedings of the eleventh international conference on Information and knowledge management*, pages 391–397, McLean, Virginia, USA, 2002. ACM.

[43] Amanda Spink, Bernard J. Jansen, Chris Blakely, and Sherry Koshman. Overlap among major web search engines. In *Proceedings of the Third International Conference on Information Technology: New Generations*, pages 370–374, Washington, DC, USA, 2006. IEEE Computer Society.

[44] Ping Wu, Ji-Rong Wen, Huan Liu, and Wei-Ying Ma. Query selection techniques for efficient crawling of structured web sources. In *Proceedings of the 22nd International Conference on Data Engineering*, page 47, Washington, DC, USA, 2006. IEEE Computer Society.

[45] Jingfang Xu, Sheng Wu, and Xing Li. Estimating collection size with logistic regression. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 789–790, Amsterdam, The Netherlands, 2007. ACM.

Appendix A

Glossary of Notations

D

The deep web data source whose size is being estimated.

N

The number of documents in D.

k_i

The number of documents respond to the $i-th$ query.

d_i

The number of unique documents returned up to the i-th query.

u_i

The number of total documents returned up to the i-th query.

OR

The overlapping rate. The fraction of u_i to k_i.

P

The fraction of documents to N.

$E(\hat{N})$

The mean value of a number of estimations.

d

A document.

q

A query.

QP

A query pool.

$M(q,D)$

A subset of D that matches q.

$M(QP,D)$

A subset of D that matches all queries in a QP.

$E(\hat{N})$

The mean value of a number of estimates.

$w(d,QP)$

The weight of d w.r.t a QP.

$w(q,QP)$

The weight of q w.r.t the QP.

$W(QP,D)$

The mean weight all the queries in the QP.

Dic

A dictionary.

t

t number of queries.

$random(t,QP)$

Randomly select t queries from the QP.

n

The number of documents checked.

Vita Auctoris

Liang Jie was born in 1984 in China. He graduated from Yu Cai Middle School in 2003 in Guangzhou, China. From there he went on to Jilin University, China where he obtained a B.Eng. in Software Engineering in 2007. He is currently a candidate for the Master's degree in Computer Science at the University of Windsor and hopes to graduate in August 2009.

Publications

- Jianguo Lu, Yan Wang, Jie Liang, Jessica Chen and Jiming Liu, "An Approach to Deep Web Crawling by Sampling", *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Volume 01*, IEEE Computer Society, 2008, pp. 718-724.

- Jie Liang, Yanyin Zhang, Jianguo Lu, Sabiha Sathulla, Ding Chen and Shaohua Wang, "A Rental Advising System Based on Service Oriented Architecture", *Proceedings of the 2008 IEEE Congress on Services - Part I - Volume 00*, IEEE Computer Society, 2008, pp. 184-190.

Conference

- 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, Sydney, Australia, September 2008.