World Wide Web (2006) 9: 585–622
DOI 10.1007/s11280-006-0227-7

Discovering Interesting Relationships among Deep Web Databases: A Source-Biased Approach

James Caverlee · Ling Liu · Daniel Rocco

Received: 8 July 2005 / Revised: 11 November 2005 / Accepted: 14 July 2006

Abstract The escalation of deep web databases has been phenomenal over the last decade, spawning a growing interest in automated discovery of interesting relationships among available deep web databases. Unlike the "surface" web of static pages, these deep web databases provide data through a web-based query interface and account for a huge portion of all web content. This paper presents a novel source-biased approach to efficiently discover interesting relationships among web-enabled databases on the deep web. Our approach supports a relationship-centric view over a collection of deep web databases through source-biased database analysis and exploration. Our source-biased approach has three unique features: First, we develop source-biased probing techniques, which allow us to determine in very few interactions whether a target database is relevant to the source database by probing the target with very precise probes. Second, we introduce source-biased relevance metrics to evaluate the relevance of deep web databases discovered, to identify interesting types of source-biased relationships for a collection of deep web databases, and to rank them accordingly. The source-biased relationships discovered not only present value-added metadata for each deep web database but can also provide direct support for personalized relationship-centric queries. Third, but not least, we also develop a performance optimization using source-biased probing with focal terms to further improve the effectiveness of the basic source-biased model. A prototype system is designed for crawling, probing, and supporting relationship-centric queries over deep web databases using the source-biased approach. Our experiments evaluate the effectiveness of the proposed source-biased analysis and discovery model, showing that the source-biased approach outperforms query-biased probing and unbiased probing.

Keywords deep web database · database ranking · probing · deep web · biased discovery

J. Caverlee (✉) · L. Liu · D. Rocco
Georgia Institute of Technology, College of Computing,
801 Atlantic Drive, Atlanta, GA 30332, USA
e-mail: caverlee@cc.gatech.edu

1 Introduction

The past few years have witnessed great strides in the accessibility and manageability of vast amounts of data—from the growth of web data resources to corporate and personal information servers. In particular, the web has seen tremendous growth of web-enabled databases that provide access to huge and growing data repositories. These web-enabled databases often provide advanced tools for searching, manipulating, and analyzing the information contained in these data repositories. Unlike the "surface" web of static pages, these "deep" web databases provide data through a web-based query interface. Recent estimates suggest that the size of this deep web greatly exceeds that of the surface web—with nearly 92,000 terabytes of data on the deep web versus only 167 terabytes on the surface web as of 2003 [23].

Deep web resources are typically under-represented on popular search engines due to the technical challenges of locating, accessing, and indexing deep web data. Since deep web data is stored in a database, traditional search engine indexers cannot discover them through the traversal of hyperlinks, but must interact with a (potentially) complex query interface. Coupled with the tremendous size of the deep web versus the surface web, the typical search engine approach of crawling and indexing pages locally faces great challenges.

Our research interest is to provide support for the management of deep web databases beyond that captured by traditional search engines. Typically, a search engine is optimized to identify a ranked list of documents (or web pages) relevant to a user query. This *document-centric* view has proven immensely successful. On the other hand, with the rise of high-quality deep web databases and the emergence of digital libraries and corporate information servers, we believe that there is ample opportunity for a new class of queries optimized not on the document level, but on the more general relationship level between deep web databases.

Rather than requesting the top-ranked documents containing a certain keyword, say "autism", we propose that a user may be more interested in *relationship-centric* queries about the many available deep web databases. For example, a user familiar with the popular online medical literature site PubMed [30] that provides access to millions of scientific and medical literature citations may be interested in posing some of the following queries:

- What other deep web databases are most similar to PubMed?
- What other deep web databases are more general than PubMed? Or more specialized?
- Are there any other deep web databases that complement PubMed's coverage?

We could also imagine extending these queries to more sophisticated ones that cover relationships among multiple deep web databases. Additionally, the granularity of the relationship discovered may be further refined to consider subsets of the databases. For example, a user may be interested in relationship-centric queries about specific journals within the PubMed database, and not PubMed as a whole.

We envision a number of scenarios in which a relationship-centric framework over deep web databases would be of significance:

- First, a relationship-centric framework can be used for supporting direct relationship queries about deep web databases like the ones listed above.

- In addition to these types of direct relationship queries, a user may simply be interested in discovering any non-obvious relationships that may exist among a group of deep web databases. The relationship-centric framework supports this type of data mining.
- The relationship-centric framework may also be used to augment traditional document-centric queries over deep web databases. For example, a user interested in medical literature may choose to query both PubMed and all databases with a similarity-based relationship to PubMed. Alternatively, a user interested in maximizing coverage of multiple topically-distinct deep web databases, may choose to query both PubMed and any other deep web database that has complementary coverage relative to PubMed.
- Finally, the relationship-centric framework may also support the refinement and generalization of traditional document-centric queries over deep web databases. A user issuing a query to PubMed may be overwhelmed by responses from a number of different medical journals. She may prefer to refine the query to the deep web databases that are more specialized on a particular topic like cancer research. Alternatively, she may find too few responses to a particular query and wish to generalize the scope of the query to include more deep web databases that have broader coverage than PubMed.

Currently, there are no effective means to answer queries that rely on a relationship-centric view of deep web databases without relying on significant human intervention or hand-tuned categorization schemes. Search engines rely on a document-centric view of the web and are not designed to handle relationship-level queries. Similarly, directory services—like the ones offered by Yahoo! and the Open Directory Project [dmoz.org]—offer general categories but do not provide coverage and specialty ratings for direct comparisons between deep web databases. So a user may find a category listing for medical literature resources that includes PubMed, but she would lack support for understanding the relationship between PubMed and the other medical literature resources, or for understanding the relationship between PubMed and resources listed under a different category.

With the rapid increase in the number and variety of available deep web databases, there is a growing need for providing a relationship-centric framework for discovering and understanding the interrelationships among deep web databases. To answer these challenges, we present a novel approach to discover interesting relationships among deep web databases based on *source-biased database analysis*. This source-biased approach supports a relationship-centric view over a collection of deep web databases through source-biased probing and source-biased relevance metrics. Source-biased database analysis and discovery presents a number of unique properties. First, our approach is capable of answering relationship-centric queries of the form posed above by focusing on the nature and degree of the relationship of one deep web database to another. Since deep web databases tend to be large and may be updated frequently [21, 23], we rely on a *sampling-based* approach to extract a portion of each deep web database through source-biased probing. Given a database like PubMed—called the *source*—the source-biased probing technique leverages the summary information of the source to generate a series of biased probes to other databases—called the *targets*. This source-biased probing allows us to determine whether a target database is relevant to the source by probing the

target with very few focused probes. Second, we introduce the *biased focus* metric to discover highly relevant deep web databases and to assess the nature of the relationship between databases. For each source database, we use the biased focus to rank target databases and to identify interesting relationship sets that support relationship-centric queries. Such relationships can be further utilized as value-added metadata for each database discovered. Third, to further reduce the number of probes necessary to assess a target database, we introduce a performance optimization called *source-biased probing with focal terms*. The main idea is to identify terms in the unbiased summary of the source that share a similar topical category and then to divide the source summary into k clusters, where each cluster represents a group of similar summary terms.

Our experiments on both simulation and web datasets show how the source-biased database analysis approach results in efficient discovery and ranking of deep web databases. We also illustrate how our approach supports relationship-centric queries through the identification of interesting relationship sets, including similarity-based and hierarchical relationship sets. Additionally, we present the design and architecture of our DynaBot system for crawling, probing, and supporting relationship-centric queries over deep web databases using the source-biased approach.

The rest of the paper is organized as follows. We present related work in Section 2 and briefly discuss the motivation and system model in Section 3. In Section 4, we describe the algorithm for source-biased analysis of databases, including source-biased probing, the biased-focus metric, and the assessment of inter-database relationships. We refine the basic source-biased algorithm with focal terms in Section 5. We present the design and architecture of the DynaBot crawler for supporting relationship-centric queries over the deep web in Section 6. In Section 7, we provide extensive experimental evidence to demonstrate the effectiveness of our algorithms and end in Section 8 with our final thoughts and notes on future enhancements.

2 Related work

The deep web (sometimes referred to as the hidden or invisible web) has garnered increased research interest in recent years. Several studies have noted the immense size of the deep web relative to the surface web of static documents [3, 9, 23]. One of the first deep web crawlers for discovering and interacting with deep web databases was proposed in [32], where the complexity of interacting with web search interfaces was noted. More recently, there have been efforts to extract data from deep web databases [7] and to match deep web query interfaces [39, 40, 43], among others.

In the database community, considerable attention has been dedicated to the *database selection* problem [5, 11–17, 22, 24, 28, 42]. In database selection, the problem is to take a query and match it to potentially relevant databases for processing. Typically the database exports a description to help guide database selection. Instead of matching a query to a set of databases, our work is concerned with analyzing and understanding the relationships among deep web databases in a source-biased framework. As we will show, these interesting relationships may be used to help guide database selection in addition to supporting relationship-centric queries.

Other researchers have previously studied the problem of *sampling* a database in an effort to generate a summary of the database internals [4, 6, 10, 17–20, 25, 37, 38]. The main purpose of these techniques is to generate a representative content summary of the underlying database in cases where the database provides only limited access. These sampling approaches typically rely on interacting with the database through a query interface and extracting sample data through a series of query probes. Querying methods suggested include the use of random queries, queries learned from a classifier, and queries based on a feedback cycle between the query and the response. In addition, Agichtein et al. [1] have developed a formal reachability graph model for assessing the quality of query-based database summarization techniques. In this paper, we show how traditional unbiased sampling approaches—especially in the context of the large size and dynamic nature of deep web databases—may be inadequate for exposing relationships among deep web databases. As a result, we promote a source-biased perspective to overcome these issues.

More recently, Ipeirotis et al. [18, 20] have introduced a probing-based approach for classifying deep web databases into a pre-determined Yahoo!-style hierarchy. However, this approach relies on a pre-learned set of queries for database classification, which requires the potentially burdensome and inflexible task of labelling training data for learning the classifier probes in the first place. Additionally, if new categories are added or old categories removed from the hierarchy, new probes must be learned and each source re-probed. Our approach is designed to work flexibly in a "bottom-up" fashion by placing each deep web database at the center of its own neighborhood of related databases.

3 System model and problem statement

In this section, we present the system model and discuss in detail current approaches for sampling deep web databases. We identify problems with the current approaches that motivate the source-biased framework for identifying interesting relationships among deep web databases. The large and increasing number of deep web databases accessible through the web not only makes a relationship-centric view over a collection of deep web databases important but also demands for an efficient and effective framework for discovering and understanding the interesting interrelationships among deep web databases.

3.1 Modeling deep web databases

We consider a *web-enabled document database* (or *deep web database*) to be a database that is composed primarily of text documents and that provides query-based access to these documents either through keyword search or more advanced search operators. In particular, we are interested in deep web databases that are beyond the control of the typical users. For example, web data sources that provide a search mechanism like the resources on the deep web, digital libraries, and databases deployed across loose corporate federations are all examples of deep web databases for which a typical user can only access through query-based mechanisms. For such types of deep web databases, relationship-centric queries of the form: "What other deep web databases are most similar to X? Or complementary to X?" would require significant human intervention to yield a satisfactory result.

We consider a universe of discourse $\mathcal{U}$ consisting of d deep web databases: $\mathcal{U} = \{D_1, D_2, \ldots, D_d\}$ where each database produces a set of documents in response to a particular query. Hence, we describe each deep web database D_i as a set of M_i documents: $D_i = \{doc_1, doc_2, \cdots, doc_{M_i}\}$. There are N terms (t_1, t_2, ..., t_N) in the universe of discourse $\mathcal{U}$, where common stopwords (like "a", "the", and so on) have been eliminated. Optionally, the set of N terms may be further refined by stemming [27] to remove prefixes and suffixes.

Adopting a vector-space model [34, 35] of the database contents, we may describe each deep web database D_i as a vector consisting of the terms in the database along with a corresponding weight:

$$\text{SUMMARY}\ (D_i) = \{(t_1, w_{i1}), (t_2, w_{i2}), \cdots, (t_N, w_{iN})\}$$

A term that does not occur in any documents in deep web database D_i will have weight 0. Typically, for any particular deep web database D_i, only a fraction of the N terms will have non-zero weight. We refer to the number of non-zero weighted terms in D_i as N_i.

This vector-based approach has received great attention and support in the information retrieval community for representing documents where it has been applied effectively across a wide variety of application settings [2]. In the database community, the vector-based approach has been popularized by GlOSS [15] and related projects.

We call the vector $\text{SUMMARY}\ (D_i)$ a *resource summary* for the deep web database D_i. A resource summary is a single aggregate vector that summarizes the overall distribution of terms in the set of documents produced by the database. To find $\text{SUMMARY}\ (D_i)$, we must first represent each document $doc_j (1 \leq j \leq M_i)$ as a vector of terms and the frequency of each term in the document:

$$doc_j = \{(t_1, freq_{j1}), (t_2, freq_{j2}), \cdots, (t_N, freq_{jN})\}$$

where $freq_{jk}$ is the frequency of occurrence of term t_k in document j. The initial weight for each term may be based on the raw frequency of the term in the document and it can be refined using alternative occurrence-based metrics like the normalized frequency of the term and the term-frequency inverse document-frequency (*TFIDF*) weight. TFIDF weights the terms in each document vector based on the characteristics of all documents in the set of documents.

Given a particular encoding for each document, we may generate the overall resource summary for each deep web database in a number of ways. Initially, the weight for each term in the resource summary may be based on the overall frequency of the term across all the documents in the database (called the *database frequency*, or *dbFreq*): $w_{ik} = dbFreq_{ik} = \sum_{j=1}^{M} freq_{jk}$. Alternatively, we can also define the weight for each term based on the number of documents in which each term occurs (called the *document count frequency, or docCount*): $w_{ik} = docCount_{ik} = \sum_{j=1}^{M} \mathcal{I}_j(t_k)$ where $\mathcal{I}_j(t_k)$ is an indicator function with value 1 if term t_k is in document j and 0 otherwise.

Once we have chosen our database model, to effectively compare two deep web databases and determine the relevance of one database to another, we need two technical components: (1) a technique for generating a resource summary; and (2) a metric for measuring the relevance between the two databases.

3.2 Estimating resource summaries

Ideally, we would have access to the complete set of documents belonging to a deep web database. We call a resource summary for D_i built on the complete set of documents an *actual resource summary* or $\text{ASUMMARY}(D_i)$. However, the enormous size of many deep web databases coupled with the non-trivial costs of collecting documents (through queries and individual document downloads) make it unreasonable to generate an actual resource summary for every deep web database available. Additionally, the well-noted dynamic nature of deep web data [21] makes extracting the complete set of documents belonging to a deep web database infeasible, since deep web databases may add new content and delete old content faster than all documents may be extracted.

As a result, previous researchers have introduced several techniques for *sampling* a database through a series of *probe queries* to generate a representative summary based on a small sample of the entire database [4, 6]. We call such a representative summary an *estimated resource summary*, denoted as

$$\text{ESUMMARY}(D_i) = \{(t_1, w_{i1}), (t_2, w_{i2}), \cdots, (t_N, w_{iN})$$

The number of occurring terms (i.e., those terms that have non-zero weight) in the estimated summary is denoted by N_i'. Typically, N_i' will be much less than the number of non-zero weighted terms N_i in the actual resource summary since only a fraction of the total documents in a database will be examined. Hence, the goal of a prober is typically to find $\text{ESUMMARY}(D_i)$ such that the relative distribution of terms closely matches the distribution of terms in $\text{ASUMMARY}(D_i)$, even though only a fraction of the total documents will be examined.

Current probing techniques for estimating resource summaries aim at estimating the overall summary of the content for a deep web database. We classify these sampling techniques into two categories: random sampling and query-based sampling.

3.2.1 Random sampling—no bias

If we had unfettered access to a deep web database, we could randomly select terms from the database to generate the estimated resource summary $\text{ESUMMARY}(D_i)$. Barring that, we could randomly select documents with which to base the estimated resource summary. We will call such a random selection mechanism an *unbiased prober* since all terms (or documents) are equally likely to be selected. In practice, an unbiased prober is unrealistic since most deep web databases only provide a query-based mechanism for extracting documents.

3.2.2 Query-based sampling—query bias

As a good approximation to unbiased probing, Callan et al. [4, 6] have introduced a query-based sampling technique for generating accurate estimates of deep web databases by examining only a fraction of the total documents. The Callan technique relies on repeatedly requesting documents from a source using a limited set of queries. Since the documents extracted are not chosen randomly, but are

biased by the querying mechanism through the ranking of returned documents and by providing incomplete access to the entire database, we say that the Callan technique displays *query bias*. There are several ways to define the limited set of queries, including random selection from a general dictionary and random selection augmented by terms drawn from the extracted documents from a database. The query bias technique has some advantages over the no bias approach, since two databases that consist of the same set of documents, but have different ranking approaches will be sampled (and hence represented by the estimated summary) differently. Through experimental validation, the Callan technique has been shown to extract high-quality resource summaries (through a number of metrics) over databases consisting of over one million pages by collecting only a few hundred documents. Since the distribution of terms across the documents of a text database follows a Zipfian distribution [4]—meaning that a few words occur in many documents, while the majority of terms occur in very few documents—extracting a small sample of documents may be successful at extracting the most popular terms in a database, resulting in a high-quality resource summary. In the rest of the paper, when we refer to an estimated resource summary $\text{ESUMMARY}(D_i)$, we mean one that has been produced by a query-biased prober.

3.2.3 Potential problems

In order to determine the relevance of one deep web database D_i to another database D_j and to assess the nature of their relationship, we require an appropriate relevance metric. There are a number of possible relevance metrics to compare two resource summaries, including a simple count of the common terms in both resource summaries to a weighted version that considers the term weights of the common terms in both resource summaries. We consider two different relevance metrics here that each emphasize a different notion of relevance.

The first is a popular symmetric relevance metric adopted from the information retrieval community for comparing the resource summaries of two databases D_i and D_j—the cosine similarity (or normalized inner product):

$$\cos(D_i, D_j) = \frac{\sum_{k=1}^{N} w_{ik} w_{jk}}{\sqrt{\sum_{k=1}^{N} (w_{ik})^2} \cdot \sqrt{\sum_{k=1}^{N} (w_{jk})^2}}$$

where w_{ik} is the weight for term k in $\text{ESUMMARY}(D_i)$ and w_{jk} is the weight for term k in ESUMMARY_{D_j}. The cosine ranges from 0 to 1, with higher scores indicating a higher degree of similarity. In contrast, the cosine between orthogonal vectors is 0, indicating that they are completely dissimilar. The cosine measures the angle between two vectors, regardless of the length of each vector.

The second relevance metric is asymmetric in nature and measures the fraction of terms in one resource summary that are also contained in the other resource summary:

$$contain(D_i, D_j) = \frac{|\text{ESUMMARY}(D_i) \cap \text{ESUMMARY}(D_j)|}{|\text{ESUMMARY}(D_i)|}$$

If all of the terms in D_i also occur in D_j, then the containment is 1. If no terms in D_i occur in D_j then the containment is 0. The containment relevance metric may be used to measure the containment of each resource summary with respect to the other, and vice versa.

We now use an example to illustrate why the existing resource summary estimation techniques are inadequate for effectively revealing interesting relationships between two databases, especially in terms of the content coverage of one (target) in the context of the other (source). We considered three real-world deep web databases—the PubMed medical literature site, the job-posting site Monster.com, and the popular search engine Google. We consider Google to be a deep web database since it provides an advanced ranking engine over a database of indexed content. We further note that all three deep web databases provide access to vast content that is constantly being updated (i.e., PubMed adds new citations, Monster provides up-to-date job postings, and Google updates its index to reflect changes in the web), meaning that a sampling-based approach is necessary to provide a current snapshot of the state of each deep web database.

Example We collected 300 documents from Google, PubMed, and Monster, respectively, using a query-based sampling technique for resource summary estimation. For each site, we issued a random query term drawn from the standard Unix dictionary and collected a maximum of four documents per query. We then extracted the plain text from each sampled document by removing all HTML tags and eliminated a list of common stopwords (e.g., "a", "the", and so on). Using the resource summaries constructed, we find that $cos(PubMed, Google) = 0.15$ and $cos(PubMed, Monster) = 0.16$. Similarly, we find that for the asymmetric relevance metric, we have $contain(PubMed, Google) = 0.19$ and $contain(PubMed, Monster) = 0.22$, meaning that 22% of the terms in the PubMed estimated summary also occur in the Google estimated summary, whereas 25% of the PubMed terms occur in Monster. Hence, for both relevance metrics we see that both Google and Monster appear to have relatively low relevance with respect to PubMed. Although Monster does provide some health-related content (like medical job postings), we expect that Google should be significantly more relevant to PubMed since it provides as much or more health-related content. When we reverse the containment calculation, we find that $contain(Google, PubMed) = 0.32$ and $contain(Monster, PubMed) = 0.36$. Interestingly, these statistics support exactly the opposite conclusion we would anticipate: that more of Google's content is contained in PubMed, than vice versa.

This example underlines a critical problem with current techniques for probing and comparing resource summaries. Current resource summary estimation techniques are concerned with generating *overall* summaries of the underlying databases. The goal is to generate essentially an unbiased estimate of the actual resource summary. Due to the Zipfian distribution of terms across the documents of a deep web database, an unbiased summary will focus on terms that are relatively popular across the space of all documents. For two deep web databases that have overlapping content (like PubMed and Google), a comparison over these unbiased summaries will not necessarily identify these specialized areas of commonality. In this example, since Google has such broad coverage, few terms in an unbiased estimated summary may be common to the PubMed estimated resource summary. Hence, many topics that are relevant for context-based database comparisons may be under-represented or overlooked completely, since the summaries contain just a

small fraction of the total terms in each database. In the context of our example, it would be interesting to discover both that Google is much more relevant to PubMed than Monster and that Google has much broader coverage than Monster. When comparing databases, the relevance metrics based on the unbiased summaries of each database like the ones described above are clearly inadequate, since they fail to adequately capture the asymmetric relationship between the two databases. This example shows both the need for a new query probing technique to hone on the common areas between two databases, allowing for more in-depth comparisons, and the need for more effective relevance metrics to more appropriately capture the nature and degree of the relationship between two databases.

4 Source-biased database analysis

Bearing these issues in mind, we propose a source-biased approach—called *source-biased database analysis*—to efficiently discover interesting relationships among text document databases. There are three fundamental steps in performing source-biased database analysis: (1) source-biased probing for deep web database discovery; (2) evaluation and ranking of discovered deep web databases with the biased focus metric; and (3) leveraging the biased perspective of sources and targets to discover interesting relationships. This framework provides the foundation for enabling relationship-centric queries over deep web databases.

4.1 Source-biased probing

In order to find the target databases that have high relevance to a source, we need to generate a source-biased summary of the target databases instead of using unbiased summaries of the targets. We propose a source-biased probing algorithm that can compute the relevance of the target databases with respect to the source in very few probes. Given a deep web database—the *source*—the source-biased probing technique leverages the summary information of the source to generate a series of biased probes for analyzing another deep web database—the *target*. This source-biased probing allows us to determine in very few interactions whether a target database is relevant to the source by probing the target with focused probes. Note that the goal of source-biased probing is *not* to generate an unbiased estimated resource summary like the query-based sampling approach discussed above. Instead, the goal is to intentionally skew the estimated summary of a target deep web database towards the source database for enabling more effective comparisons. This source-biasing is especially important for supporting relationship-centric queries over a diverse and large set of deep web databases for which exhaustive sampling of each database is infeasible.

To help differentiate the source-biased approach from others discussed in Section 3, in this section we use σ to denote the source database and τ to denote the target database instead of D_i and D_j. Given two databases σ and τ, the output of the source-biased probing is a subjective resource summary for τ that is biased towards σ. We denote the source-biased summary of the target database as:

$$\text{ESUMMARY}_\sigma(\tau) = \{(t_1, w_1^\sigma), (t_2, w_2^\sigma), ..., (t_N, w_N^\sigma)\}$$

N is the total number of terms used in analyzing the set of deep web databases. w_i^σ $(1 \leq i \leq N)$ is the weight of term t_i, defined using one of the weight functions introduced in Section 3.1. It is important to note that typically the inequality $w_j \neq w_j^\sigma$ does hold.

Concretely, the source-biased probing algorithm generates a source-biased summary for a target as follows: It begins with an unbiased resource summary of the source σ, denoted by $\text{ESUMMARY}(\sigma)$, that is generated through a standard application of query-based sampling. It uses the unbiased resource summary $\text{ESUMMARY}(\sigma)$ as a dictionary of candidate probe terms and sends a series of probe terms, selected from $\text{ESUMMARY}(\sigma)$, to the target database τ; for each probe term, it retrieves the top m matched documents from τ, generates summary terms and updates $\text{ESUMMARY}_\sigma(\tau)$, the source-biased summary of the target database. Note that the updating process requires the simple updating of the term-frequency statistics in the source-biased summary based on the statistics extracted from the new batch of sampled documents. This process repeats until a stopping condition is met. Figure 1 illustrates the source-biased probing process. Initially we constrain the source-biased probes to keyword-based probes, though we anticipate extending this coverage in the future.

In general, we may extend the pairwise source-biased probing algorithm along two dimensions: to consider k bias sources with one target database, or to consider k target databases with one source.

Let $\mathcal{S}$ be a subset of databases from the universe of discourse $\mathcal{U}$. $\mathcal{S}$ consists of k $(0 \leq k \leq d)$ databases from $\mathcal{U}$, where $\mathcal{S} = \{\sigma_1, \sigma_2, \ldots, \sigma_k\}$. Each $\sigma_j \in \mathcal{S}$ corresponds to a database $D_i \in \mathcal{U}$. We may then define a set of biased summary estimates for σ_j based on the sources of bias in $\mathcal{S}$: $\{\text{ESUMMARY}_{\sigma_1}(\tau), \text{ESUMMARY}_{\sigma_2}(\tau), \ldots \text{ESUMMARY}_{\sigma_k}(\tau)\}$. Hence, with source-biased probing, a target database may be viewed through the lens of a set of biasing sources, rather than through a single unbiased summary. As a result, we may use these biased summaries to characterize the target database, which can be helpful for clustering and categorization applications, as well as for supporting query routing to the appropriate deep web database. For example, we may discover that the PubMed-biased summary of Monster contains many more medical-related job postings than a PubMed-biased summary of the technical jobs site Dice. Hence, a query engine could choose to route medical-jobs-related queries to Monster, rather than Dice.

SourceBiasedProbing(Database σ, Database τ)
For target database τ, initialize $\text{ESUMMARY}_\sigma(\tau) = \emptyset$.
repeat
Invoke the probe term selection algorithm to select a one-term query probe q from the source of bias $\text{ESUMMARY}(\sigma)$.
Send the query q to the target database τ.
Retrieve the top-m documents from τ.
Update $\text{ESUMMARY}_\sigma(\tau)$ with the terms and frequencies from the top-m documents.
until Stop probing condition is met.
return $\text{ESUMMARY}_\sigma(\tau)$

Figure 1 Source-biased probing algorithm.

Similarly, a set of target databases can be evaluated and compared with respect to one source using the source-biased lens. Extending the pairwise source-biased probing algorithm along this dimension allows for the evaluation and ranking of the target databases with respect to the source database. We discuss this type of analysis in great detail in the following section.

The performance and effectiveness of the source-biased probing algorithm depends upon a number of factors, including the network characteristics of the target database (like uptime, latency, etc.), the ranking algorithm used at the target database for ranking the returned documents, the selection criterion used for choosing source-specific candidate probe terms, and the type of stop condition used to terminate the probing process. In this paper, we focus our attention on the selection criterion and the stop probing condition.

4.1.1 Mechanisms to select probe terms

There are several possible ways to select the probes based on the statistics stored with each resource summary, including uniform random selection and selection based on top-weighted terms. In general, the selection criterion will recommend a query term drawn from the set of all non-zero weighted terms in the unbiased source summary ESUMMARY(σ).

Uniform Random Selection In this simplest of selection techniques, each term that occurs in ESUMMARY(σ) has an equal probability of being selected, i.e., $Prob(\text{selecting term } j) = \frac{1}{N_\sigma}$.

Weight-Based Selection Rather than randomly selecting query terms, we could instead rely on a ranking of the terms by one of the statistics that are recorded with each resource summary. For example, all terms in ESUMMARY(σ) could be ranked according to the weight of each term. Terms would then be selected in descending order of weight. Depending on the type of weight cataloged (e.g., *dbFreq*, *docCount*, etc.), several flavors of weight-based selection may be considered.

4.1.2 Different types of stop probing conditions

The stop probing condition is the second critical component in the source-biased probing algorithm. It is interesting to note that the more documents the source-biased probing algorithm extracts from a target database, the more likely the source-biased estimated summary of the target database will tend to closely correlate with the unbiased estimated summary of the target database, meaning that the choice of stop probing condition is vitally important. So long as there are still query terms available in the source summary for use in probing, the source-biased probing algorithm will continue to extract documents, even if the queries issued are less indicative of the source subject matter. For example, if we are using PubMed as a source of bias for probing a non-relevant sports database, a poorly selected stop probing condition could result in the algorithm exhausting all of the heavily-weighted scientific and medical query probes, forcing the algorithm to send lowly-weighted (and hence, less indicative of PubMed) probes, resulting in a target summary that closely resembles an unbiased summary. Hence, the stop probing

condition is critical to guide the quality of the source-biased probing algorithm. We consider four different types of conditions that might be used in practice:

Number of Queries After some fixed number of query probes (*MaxProbes*), end the probing. This condition is indifferent to the number of documents that are examined for each database.

Documents Returned In contrast to the first technique, the second condition considers not the number of queries, but the total number of documents (*MaxDocs*) returned by the database.

Document Thresholding Rather than treating each document the same, this third alternative applies a threshold value to each document to determine if it should be counted toward *MaxDocs*. For each document, we may calculate the relevance of the document to the source of bias ESUMMARY(σ). If the document relevance is greater than some threshold value, then the document is counted. Otherwise, the document is discarded.

Steady-State Rather than relying on a count of queries or documents, this final stopping condition alternative instead relies on the estimated summary reaching a steady-state. After each probe, we calculate the difference between the new value of ESUMMARY$_\sigma(\tau)$ and the old value. If the difference (which may be calculated in a number of ways) is less than some small value ϵ, then we consider the summary stable and stop the probing.

4.2 Evaluating and ranking databases with biased focus

Given a source and a target database, once we generate the source-biased summary for the target database, we need an efficient mechanism to measure the source-biased relevance of a target database with respect to the source. Once a set of target databases have been evaluated with the source-biased relevance metric, we can then rank the targets with respect to the source of bias. We perform this task using the second component of source-biased database analysis—a source-biased metric.

Let σ denote a source database modeled by an unbiased summary and τ denote a target database with a σ-biased summary, and let $focus_\sigma(\tau)$ denote the source-biased focus measure. We define $focus_\sigma(\tau)$ to be a measure of the topical focus of the target database τ with respect to the source of bias σ. The focus metric ranges from 0 to 1, with lower values indicating less focus and higher values indicating more focus. In general, *focus* is not a symmetric relation. We may describe any two deep web databases σ and τ with the focus in terms of σ by $focus_\sigma(\tau)$ or in terms of τ by $focus_\tau(\sigma)$. The biased focus is intended as a measure of *inclusion* [26]; that is, it measures the amount of the source included in the target.

There are several ways to calculate the biased focus for a source and a target. Adopting the cosine similarity introduced in the previous section for the case of a source-biased target summary, we may define the cosine-based focus as:

$$Cosine\_focus_\sigma(\tau) = \frac{\sum_{k=1}^{N} w_{\sigma k} w_{\tau k}^{\sigma}}{\sqrt{\sum_{k=1}^{N} (w_{\sigma k})^2} \cdot \sqrt{\sum_{k=1}^{N} \left(w_{\tau k}^{\sigma}\right)^2}}$$

where $w_{\sigma k}$ is the weight for term k in ESUMMARY(σ) and $w^{\sigma}_{\tau k}$ is the σ-biased weight for term k in ESUMMARY$_{\sigma}(\tau)$. Again, we note that the cosine ranges from 0 to 1, with higher scores indicating a higher degree of similarity.

Alternatively, we could approximate the focus measure by computing the ratio of common terms between source and target over the source summary estimate. We call this method the *common-term based focus measure*, denoted by $CT\ focus_{\sigma}(\tau)$.

$$CT\ focus_{\sigma}(\tau) = \frac{|\text{ESUMMARY}(\sigma) \cap \text{ESUMMARY}_{\sigma}(\tau)|}{|\text{ESUMMARY}(\sigma)|}$$

This approximation counts the number of common terms between the source of bias and the target and divides by the size of the source of bias. So if all terms in the source of bias occur in the target, then the target is perfectly focused on the source and $CT\ focus_{\sigma}(\tau) = 1$. Conversely, if no terms in the source of bias occur in the target, then the target has no focus on the source and $CT\ focus_{\sigma}(\tau) = 0$. Unfortunately, a common-term based focus measure will tend to understate the importance of highly-weighted terms and overvalue the importance of lowly-weighted terms. An obvious solution to address the above-mentioned problem is to use the *term-weight based focus measure*, denoted by $TW\ focus_{\sigma}(\tau)$:

$$TW\ focus_{\sigma}(\tau) = \frac{\sum_{k \in \text{ESUMMARY}_{\sigma}(\tau)} w_{\sigma k}}{\sum_{k \in \text{ESUMMARY}(\sigma)} w_{\sigma k}}$$

where $w_{\sigma k}$ is the weight for term k in ESUMMARY$_{\sigma}$. The term weight based focus measure can be seen as a generalization of the ctf_{ratio} introduced in [6].[1]

While the $TW\ focus_{\sigma}(\tau)$ approximation overcomes the problems of the $CT\ focus_{\sigma}(\tau)$, it introduces new issues. For example, the term weights used in the $TW\ focus_{\sigma}(\tau)$ approximation are from the unbiased summary of the source. Thus the actual weights of terms in the source-biased estimate may be distorted by relying on the unbiased summary weights.

Intuitively, the cosine-based biased focus is the most appealing of the three biased focus candidates since it seems to more reasonably capture the relevance between two deep web databases. In the experiments section we show that, compared with $TW\ focus_{\sigma}(\tau)$ and $CT\ focus_{\sigma}(\tau)$, the $Cosine\_focus_{\sigma}(\tau)$ measure can quickly approximate the actual focus measure using fewer documents.

4.2.1 Ranking relevant databases

Given an appropriate biased focus measure, we may probe a group of target databases to identify the most relevant databases to the source of bias. For a single source of bias D_1 from our universe of discourse $\mathcal{U}$, we may evaluate multiple target databases D_2, D_3, ..., D_d. For each target database, we may evaluate the appropriate focus measure for each source–target pair (i.e., $focus_{D_1}(D_2)$, $focus_{D_1}(D_3)$, etc.). We may then rank the target databases in descending order in terms of their source-biased focus with respect to D_1.

[1] The ctf_{ratio} is presented in the context of comparing an estimated resource summary DB' to an actual resource summary DB. $ctf_{ratio} = \sum_{i \in DB'} ctf_i / \sum_{i \in DB} ctf_i$, where ctf_i = number of times term i occurs in the source. Here, we have generalized this formulation for comparison of summaries from different databases, and for use with term weightings other than the ctf.

4.3 Identifying interesting inter-database relationships

The critical third component of source-biased database analysis are the techniques for exploiting and understanding relationships between deep web databases using a source-biased lens. By analyzing the nature of the relationships between deep web databases, we will provide support for relationship-centric queries. For example, we may identify relationship sets for a source that support queries of the form: "What other deep web databases are most similar to X? Or complementary to X?", among others.

As we have discussed before, current search and directory technologies for comparing deep web databases (such as search engines or Yahoo!-like directories) are not optimized for the type of relationship-centric queries that have a strong source-biased flavor. Some examples were given in the introduction of this paper. In contrast, our source-biased probing framework and biased focus measure provide the flexible building blocks for automated identification of interesting relationships between deep web databases, especially since the framework promotes an asymmetric source-biased view for any two deep web databases. Our relationship discovery module creates a flexible organization of deep web databases, where each database is annotated with a list of relationship sets. The two typical relationship types we have identified are similarity-based and hierarchical-based.

4.3.1 Similarity-based relationships

Given the universe of discourse $\mathcal{U} = \{D_1, D_2, \ldots, D_d\}$, we identify three similarity-based relationship sets for a particular deep web database D_i. These relationship sets are defined in terms of threshold values λ_{high} and λ_{low}, where $0 \leq \lambda_{low} \leq \lambda_{high} < 1$.

$\lambda -$ **equivalent** The first relationship says that if both $focus_{D_i}(D_j) > \lambda_{high}$ and $focus_{D_j}(D_i) > \lambda_{high}$ hold, then we may conclude that D_i is sufficiently focused on D_j and D_j is sufficiently focused on D_i. Hence, the two databases are approximately the same in terms of their content coverage. We call this approximate equality λ-equivalence. It indicates that the equivalence is not absolute but is a function of the parameter λ_{high}. Formally, $\lambda\text{-}equivalent(D_i) = \{\forall D_j \in \mathcal{U} | focus_{D_i}(D_j) > \lambda_{high} \wedge focus_{D_j}(D_i) > \lambda_{high}\}$.

$\lambda -$ **mutex** If both $focus_{D_i}(D_j) < \lambda_{low}$ and $focus_{D_j}(D_i) < \lambda_{low}$ hold, then we can conclude that D_i and D_j are sufficiently concerned with different topics since neither one is very focused on the other. We annotate this approximately mutually exclusive (mutex) nature with the λ prefix. Formally, $\lambda\text{-}mutex(D_i) = \{\forall D_j \in \mathcal{U} |\ focus_{D_i}(D_j) < \lambda_{low} \wedge focus_{D_j}(D_i) < \lambda_{low}\}$.

$\lambda -$ **overlap** When two deep web databases D_i and D_j are neither λ-equivalent nor λ-mutex, we say that the two deep web databases λ-overlap. Formally, $\lambda\text{-}overlap(D_i) = \{\forall D_j \in \mathcal{U} | D_j \notin \lambda\text{-}mutex(D_i) \wedge D_j \notin \lambda\text{-}equivalent(D_i)\}$.

4.3.2 Hierarchical relationships

In addition to similarity-based relationship sets, we also define hierarchical relationship sets by measuring the relative coverage of target databases in $\mathcal{U}$ with

respect to a particular deep web database D_i (source). These hierarchical relationship sets are defined in terms of a parameter λ_{diff}, where $0 \leq \lambda_{diff} \leq 1$.

$\lambda -$ **superset** If $focus_{D_i}(D_j) - focus_{D_j}(D_i) > \lambda_{diff}$, then a relatively significant portion of D_i is contained in D_j, indicating that D_j has a λ-superset relationship with D_i. We use the λ prefix to indicate that D_j is not a strict superset of D_i, but rather that the relationship is parameterized by λ_{diff}. Formally, $\lambda\text{-}superset(D_i) = \{\forall D_j \in \mathcal{U} | focus_{D_i}(D_j) - focus_{D_j}(D_i) > \lambda_{diff}\}$.

$\lambda -$ **subset** Conversely, If $focus_{D_j}(D_i) - focus_{D_i}(D_j) > \lambda_{diff}$, then a relatively significant portion of D_j is contained in D_i, indicating that D_j has a λ-subset relationship with D_i. Similarly, D_j is not a strict subset of D_i, but rather the relationship is parameterized by λ_{diff}. Formally, $\lambda\text{-}subset(D_i) = \{\forall D_j \in \mathcal{U} | focus_{D_j}(D_i) - focus_{D_i}(D_j) > \lambda_{diff}\}$.

We note that the determination of the appropriate λ-values is critical for the correct assignation of databases to each relationship set. In our experiments section, we illustrate how these relationship sets may be created; for now, we leave the optimization of λ-values as future work.

4.3.3 Using relationship sets

Both similarity-based and hierarchy-based inter-database relationships can be generated automatically, and used as metadata annotation to each of the deep web databases. These source-biased relevance data provide a flexible foundation for relationship analysis among deep web databases. For any deep web database D_i, we need only consult the appropriate relationship set to evaluate a relationship-centric query. The three similarity-based relationship sets provide the basis for answering queries of the form: "What other databases are most like X? Somewhat like X? Or complementary to X?". The two hierarchical-based sets provide the basis for answering queries of the form: "What other databases are more general than X? Or more specialized than X?". Of course, the relationship-centric queries may be further refined by considering a number of criteria besides topical relevance, including trust, quality-of-service, and size, among others.

In addition, these relationship sets are useful for routing regular document-centric queries to appropriate databases. For example, a user interested in medical literature may choose to query both PubMed and all of the databases that have a λ-*equivalence* relationship with PubMed. Alternatively, a user interested in maximizing coverage of multiple topically-distinct deep web databases, may choose to query both the source database she knows about and any members in the mutually exclusive set of the source database. The hierarchical relationship sets are particularly helpful in cases where a user may refine a query to more specialized resources, or alternatively, may choose to generalize the scope of the query by considering databases further up the hierarchy to get more matching answers. In this paper, our goal is to illustrate the importance of relationship sets and show that they may be discovered using source-biased probing. We anticipate the further exploration of relationship sets in our future work.

5 Focal term probing

One of the critical parameters to the success of source-biased probing is the choice of probe terms from the source of bias σ. We have discussed several selection techniques as well as different ways to define stop-probing conditions. In this section we propose a refinement over these simple selection techniques whereby the source summary is segmented into k groups of co-occurring terms. The main idea is to iteratively select one term from each of the k groups to probe the target. We call these terms the focal terms of the corresponding group. When used in conjunction with the general source-biased probing algorithm, we have an enhanced version called *source-biased probing with focal terms*. Like the basic algorithm of source-biased probing, the goal remains to produce source-biased target resource summaries that are effective for detecting interesting relationships between a source of bias and a target. A unique advantage of using focal terms is that these source-biased summaries of target databases can be generated in far fewer queries and with higher quality.

5.1 Focal terms and focal term groups

Let σ denote a source database with its unbiased resource summary ESUMMARY_σ. We denote the set of terms with non-zero weight in ESUMMARY_σ (i.e., the terms that actually occur in the database σ) as $Terms(\sigma)$, where $Terms(\sigma)$ consists of n terms t_1, t_2, ..., t_n. A *focal term group* is a subset of terms in the set $Terms(\sigma)$ that co-occur in the documents of σ. We denote a focal term group i as $FTerms_i$. The main idea behind source-biased probing with focal terms is to partition the set $Terms(\sigma)$ into k disjoint term groups such that the terms within each term group co-occur in documents of σ more frequently than they do with terms from other term groups. We note this measure of co-occurrence is rather coarse; our notion of co-occurrence merely indicates that two words occur in the same document together, regardless of their semantic relationship. Recall that in the vector-space model adopted in this paper, the order of words within a document and the nearness of one word to another in a document are not considered, though we anticipate refining the measure of co-occurrence in future work.

Formally, we need an algorithm that can find a partition of $Terms(\sigma)$ into k focal term groups:

$$Terms(\sigma) = \{FTerms_1, \cdots, FTerms_i, \cdots, FTerms_k | \bigcup_{i=1}^{k} FTerms_i$$

$$= \{t_1, ..., t_n\} \text{ and } FTerms_i \cap FTerms_j = \emptyset\}$$

In Table 1, we show an example of five focal term groups for a collection of 100 PubMed documents. Note that k is intended to be very small since the focal term groups are meant to be very coarse. We will describe the concrete algorithm to find k partitions of the set $Terms(\sigma)$ in the next section.

Given k focal term groups, by selecting a focal term from each term group $FTerms_i$ as a probing query, we hope to retrieve documents that also contain many of the other words in that focal term group. For example, suppose we are using a frequency-based measure for query probe selection from PubMed. The top four

query terms may be "brain", "gene", "protein", and "nucleotide". Suppose these four terms tend to co-occur with each other as indicated in Table 1. By sending the first query "brain" to a target database, we could reasonably expect to find the other three terms since our analysis of the source indicates that these four terms tend to co-occur. A naive source-biased prober would ignore this co-occurrence information and, instead, send the other three queries "gene", "protein", and "nucleotide", even though we might reasonably expect for those queries to generate documents similar to the first query "brain". In essence, we will have used four queries when a single query would have sufficed at adequately exploring the term space of the target. In cases in which both the source and target database have similar term co-occurrences, then we would anticipate focal term probing providing an advantage over the other probe selection techniques.

The sophistication of source-biased probing with focal terms is to identify these co-occurrence relationships in order to reduce the number of queries necessary to efficiently detect relationships between a source and a target database. By using focal terms, we may generate more accurate biased summaries of target databases in far fewer probe queries and with higher quality.

In an ideal case, every focal term group would consist of terms that only co-occur with each other and not with any other terms in the other focal terms groups. By selecting a single term from each perfectly segmented term group, we ideally could send no more than k probes, one for each focal term group. Each probe would produce a document that contained every other term in that focal term group. In the more realistic setting, we will need to handle varying degrees of co-occurrence, but we still expect a good reduction in the number of probes necessary to generate a high-quality biased summary estimate for each target database.

It is important to note that, unlike previous research in grouping terms—for query-expansion [31, 41] or finding similar terms [36]—our goal is not to find close semantic relationships between terms, but rather to find very coarse co-occurrence associations among terms to support a more efficient and effective biased resource summary estimation. For example, though we may discover that "brain" and "protein" tend to co-occur, we do not claim that there is a close semantic relationship between the two terms.

5.2 Finding focal terms

Now that we have discussed the motivation of finding focal terms, we are still faced with the task of actually segmenting $Terms(\sigma)$ into k groups of focal terms. In this section, we discuss how we may adapt a popular clustering technique to the problem of focal term discovery. Recall that in Section 3.1, we view a deep web database D_i

Table 1 Example focal terms for PubMed.

Group	Terms
1	Care, education, family, management, ...
2	Brain, gene, protein, nucleotide, ...
3	Clinical, noteworthy, taxonomy, ...
4	Experimental, molecular, therapy, ...
5	Aids, evidence, research, winter, ...

as a set of documents, each of which is described by a vector of terms and weights. We now invert our view of a database using the same set of information. We consider a database D_i as a collection of *terms*, each of which is described by a vector of the documents in which the term occurs and a weight describing the occurrence frequency of the term in the corresponding document. Hence, we have: $Terms(D_i) = \{term_1, term_2, \cdots, term_N\}$.

For the N terms in the database, each $term_j (1 \leq j \leq N)$ is a vector of documents and weights:

$$term_j = \{(doc_1, w_{j1}), (doc_2, w_{j2}), \cdots, (doc_M, w_{jM})\}$$

We can define a segmentation technique for finding focal term groups by clustering the set $Terms(D_i)$ into k clusters. Given the term vectors and the similarity function, a number of clustering algorithms can be applied to partition the set $Terms(D_i)$ of N terms into k clusters. We choose Simple K-Means since it is conceptually simple and computationally efficient. The algorithm starts by generating k random cluster centers. Each term is assigned to the cluster with the most similar (or least distant) center. The similarity is computed based on the closeness of the term and each of the cluster centers. Then the algorithm refines the k cluster centers based on the centroid of each cluster. Terms are then re-assigned to the cluster with the most similar center. The cycle of calculating centroids and assigning terms in $Terms(D_i)$ to k clusters repeats until the cluster centroids stabilize. Let C denote a cluster in the form of a set of terms in the cluster. The centroid of cluster C is:

$$centroid_C = \left\{ \begin{array}{c} (doc_1, \frac{1}{|C|} \sum\limits_{j \in C} w_{j1}) \\ (doc_2, \frac{1}{|C|} \sum\limits_{j \in C} w_{j2}) \\ \cdots \\ (doc_M, \frac{1}{|C|} \sum\limits_{j \in C} w_{jM}) \end{array} \right\}$$

where w_{jl} is the weight of term j in document l, and the formula $\frac{1}{|C|} \sum_{l \in C} w_{jl}$ denotes the average weight of the document l in the cluster C. A sketch of the K-Means term clustering based on term-vector of a deep web database is provided in Figure 2.

The similarity function used in Figure 2 can be defined using a number of functions. In this paper, we use the cosine similarity function. Given a set of N terms and a set of M documents, where w_{ik} denotes the weight for term k in document i $(1 \leq k \leq N, 1 \leq i \leq M)$, the cosine function prescribes:

$$sim(term_i, term_j) = \frac{\sum\limits_{k=1}^{N} w_{ik} w_{jk}}{\sqrt{\sum\limits_{k=1}^{N} (w_{ik})^2} \cdot \sqrt{\sum\limits_{k=1}^{N} (w_{jk})^2}}$$

In Section 7 we report the initial experiments on effectiveness of using focal terms to optimize the source-biased probing algorithm, showing that the source-biased algorithm with focal terms results in more efficient probing for varying numbers of focal-term groups (Figure 8).

FocalTerms(Number of Clusters k, Input Vectors $\mathcal{D}$)
Let $\mathcal{D} = \{d_1, ..., d_n\}$ denote the set of n term vectors
Let M denote the total number of documents in $\mathcal{D}$
Let $d_j = < (doc_1, w_j1), \ldots, (doc_M, w_{jM}) >$ denote a term vector of M elements, w_{jl} is the TFIDF weight of the doc_l in term j $(l = 1, \ldots, M)$
Let $\mathcal{C} = \{C_1, ..., C_k\}$ denote a clustering of $\mathcal{D}$ into k clusters.
Let μ_i denote the center of cluster C_i
foreach cluster C_i
  Randomly pick a term vector, say d_j from $\mathcal{D}$
  Initialize a cluster center $\mu_i = d_j$, where $d_j \in \mathcal{D}$
repeat
  foreach input term vector $d_j \in \mathcal{D}$
    foreach cluster $C_i \in \mathcal{C}$ $i = 1, \ldots, k$
      compute $\delta_i = sim(d_j, \mu_i)$
    if δ_h is the smallest among $\delta_1, \delta_2, \ldots, \delta_k$
      μ_h is the nearest cluster center to d_j
    Assign d_j to the cluster C_h
  // refine cluster centers using centroids
  foreach cluster $C_i \in \mathcal{C}$
    foreach doc l in d_j $(l = 1, \ldots, M))$
      $cw_{ij} \leftarrow \frac{1}{|C_i|} \sum_{l=1}^{M} w_{jl}$
    $\mu_i \leftarrow < (doc_1, cw_{i1}), \ldots, (doc_M, cw_{iM}) >$
until cluster centers no longer change
return $\mathcal{C}$

Figure 2 Focal term clustering algorithm.

5.3 Selecting focal-based probes

Once the k focal term groups have been constructed for a source, the remaining problem is how to select the best terms for probing a target database. We propose a simple round-robin selection technique whereby a single term is selected from each focal term group in turn. In each round, a single term may be selected according to one of the probe selection techniques discussed above, like uniform selection or weighted selection. Once a single term has been selected from each group, the cycle repeats by selecting a second term from each group, a third term, and so on. The

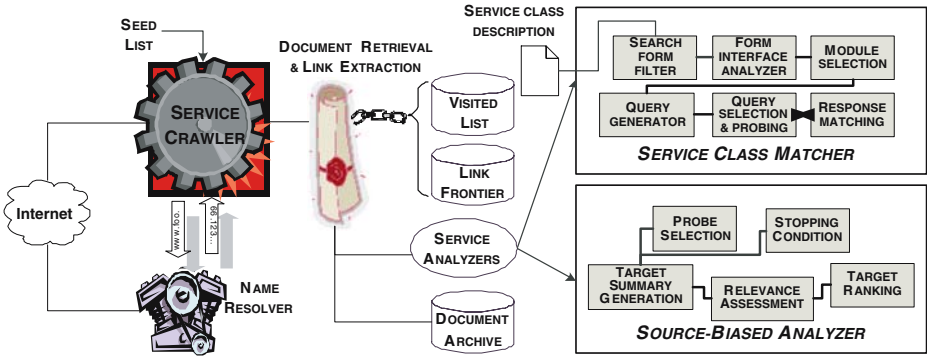

Figure 3 DynaBot system architecture.

cycle of selecting focal terms and querying the target database may be stopped according to one of the stop probing conditions discussed above. Given this basic strategy, we may use a number of techniques for determining the order by which to select terms from the k groups and for selecting probe terms from each focal term group. One way to determine the order of focal term groups is based upon the size of each group. We begin with the group with the most terms and end each cycle with the group that has the smallest number of terms.

6 Implementing source-biased database analysis in DynaBot

In this section, we present the preliminary design and architecture of the DynaBot system for supporting relationship-centric queries over the deep web. The deep web of online web-enabled databases is a large and growing component of the web—with recent estimates suggesting that there are nearly 92,000 terabytes of data on the deep web versus only 167 terabytes on the surface web [23]. Most existing web crawlers tend to ignore the data offered by deep web databases due to the technical difficulties of locating, accessing, and indexing deep web data. Traditional crawling and indexing techniques that have shown tremendous success on the surface web of hyperlinked pages are insufficient for the deep web—where data is stored in databases or produced in real-time in response to a particular user query.

In response to these challenges, we have been developing the DynaBot system for harnessing the vast amounts of data available in web-enabled databases [33]. DynaBot is designed with a modular architecture to support the entire lifecycle of supporting deep web data access, ranking, and query support—including the relationship-centric queries introduced in this paper. Figure 3 presents the overall architecture of DynaBot, with an emphasis on the initial crawling, discovery, and source-biased analysis modules.

DynaBot utilizes an advanced crawler architecture that includes standard crawler components like a URL frontier manager, network interaction modules, global storage and associated data managers, and document processors, as well as the pluggable DynaBot-specific semantic analyzers, which analyze the candidate deep web databases. We note that the name resolver in Figure 3 takes a URL and converts it into the corresponding IP address. The current semantic analyzers that have been incorporated into DynaBot include the *service class matcher* and the *source-biased analyzer*.

Rather than rely on a generic crawling strategy, we use the focused crawling framework to guide DynaBot on domain-specific crawls for identifying deep web databases related to a specific topic. Focused crawling has previously been introduced for guiding traditional web crawlers to web pages related to a specific topic [8]. A DynaBot crawl begins from a seed list of URLs that may be supplied by the user and geared toward a particular domain. So a DynaBot crawl for PubMed-related resources could begin from a URL seed list including PubMed and URLs that point to PubMed. The *service class matcher* uses focused crawling of the deep web to discover candidate deep web databases that are relevant to a specific domain of interest—e.g., bioinformatics sources, online retailers, etc. The main idea of the service class matcher is to support guided matching of candidate deep web databases by finding members of a common *service class* of functionally similar deep web

databases. Each service class is encoded in a service class description that describes the key attributes that define the service class. When the crawler comes across a web-based query interface, it invokes the service class matcher to determine if it is indeed a candidate of the service class. The service class matcher includes tools for probing a candidate deep web site, generating a site-specific script for interacting with a deep web database, and classifying a site as a member of a particular service class based on the service class description. The classification component considers the input schema of the deep web database's query interface, the output schema of sampled data, and a seed list of example probing templates for the particular service class. The output of the *service class matcher* is a set of functionally similar deep web databases for use by the *source-biased analyzer*. For the PubMed example, the deep web databases identified would all be members of a service class for medical and scientific literature databases. The DynaBot crawling module may be run for multiple service class instances to discover a large and diverse set of deep web databases for consideration by the subsequent modules.

The *source-biased analyzer* module uses the site-specific scripts generated in the service class matching module for interacting with each deep web database discovered. The source-biased probing, biased focus evaluation, and relationship-set discovery introduced in this paper are all incorporated into the source-biased analyzer module of DynaBot. Our current efforts are focused on continuing to enhance the capability and efficiency of these two modules, as well as incorporating additional semantic analyzers for enhanced deep web discovery.

7 Experiments

In this section, we describe five sets of experiments designed to evaluate the benefits and costs of our source-biased approach compared to existing approaches. The first set of experiments intends to show the effectiveness of our source-biased probing algorithm and performance comparison with query probing and unbiased probing. The second set evaluates the biased focus measure for ranking deep web database. The third set is designed to show the efficiency of the biased focus measure in identifying interesting inter-database relationships. The fourth set of experiments evaluates the efficacy of source-biased probing with focal terms by comparing the basic source-biased probing versus source-biased probing with varying number of groups of focal terms. Our experiments show that source-biased probing with focal terms can achieve about 10% performance improvement over the basic algorithm for source-biased probing. And the final set of experiments considers the impact of several key parameters on the overall performance of the source-biased approach.

We choose two different sets of deep web databases for our experiments: (1) a large collection of newsgroups designed to emulate the diversity and scope of real-world deep web databases; and (2) a modest collection of real-world deep web deep web databases. Since the contents of deep web databases in the deep web collection change frequently and are beyond our control, and in an effort not to overload any one site, we relied on the newsgroup dataset for rigorous experimental validation. We additionally note that a similar newsgroup setup has been used before to emulate deep web databases [18].

Newsgroup Collection We collected articles from 1,000 randomly selected usenet newsgroups over the period June to July 2003. We eliminated overly small newsgroups containing fewer than 100 articles, heavily spammed newsgroups (which have a disproportionate number of off-topic messages), and newsgroups with primarily binary data. After filtering out these groups, we were left with 590 *single topic* newsgroups, ranging in size from 100 to 16,000 articles. In an effort to match the heterogeneity and scope inherent in many real-world deep web databases, we constructed 135 additional groups of *mixed topics* by randomly selecting articles from anywhere from 4 to 80 single topic newsgroups, and 55 *aggregate topic* newsgroups by combining articles from related newsgroups (e.g., by selecting random documents from all the subgroups in comp.unix.* into a single aggregate group). In total, the newsgroup collection consists of over 2.5 GB worth of articles in 780 groups.

Deep Web Collection For the second collection, we randomly selected 50 sites from the ProFusion [29] directory of deep web sites, in addition to Google and PubMed. We queried each site with a randomized set of single-word probes drawn from the standard Unix dictionary, and collected a maximum of 50 documents per site. Previous research has indicated that small samples of several hundred pages may result in high quality resource summaries over databases consisting of over one million pages, since the distribution of terms across the documents of a text database follows a Zipfian distribution [4]. In this case, we choose a sample size of 50 documents to determine if even smaller samples can yield positive results in the source-biased context.

Probing Framework We built a probing engine in Java 1.4 for use in all of our experiments. For each group in both datasets, we constructed the actual resource summary based on the overall term frequency of each term (*dbFreq*). We eliminated a set of common stopwords (e.g., “a”, “the”, and so on) as well as collection-specific stopwords (e.g., “wrote”, “said”, and so on for the newsgroup collection). Terms were not stemmed.

7.1 Effectiveness of source-biased probing

The goal of our first set of experiments is to compare source-biased probing with existing probing techniques such as query probing and unbiased probing and to evaluate the efficiency and quality of source-biased probing. The source-biased probing shows significant gain in terms of the percentage of documents probed that are similar to the source. For this experiment, we assume that the document download costs outweigh the query issuing cost. Hence we evaluate the efficiency of source-biased probing in terms of the number of documents required to be extracted from each target and the percentage of the documents extracted that are similar to the source. The higher percentage of documents similar (relevant) to the source, the more effective a probing algorithm is.

We selected 100 random source–target pairs from the newsgroup collection. For each pair, we evaluated four probing techniques—a source-biased prober (*Source Bias*) that selects probe terms from the source summary in decreasing order of *dbFreq*; a query-biased prober (*Query Bias 1*) that randomly selects probes from the

standard Unix dictionary of English terms; a query-biased prober (*Query Bias 2*) that selects its initial probe from the Unix dictionary, but once the first document has been retrieved from the target, all subsequent probes are selected based on the estimated *dbFreq* of the target's resource summary; and an unbiased prober (*No Bias*) that selects documents at random from each target. For each pair, we evaluated each of the four probing techniques for up to 100 total documents extracted from each target, collecting a maximum of five documents per probe query from each target.

In Figure 4, we show the average percentage of documents similar (relevant) to the source ($Cosine\_focus_{\sigma}(\tau)$) over all 100 source–target pairs as a function of the number of documents examined in each target. The percentage of the documents extracted that are similar to the source (biased *focus* measure) indicates the quality of document being extracted from each target. We see that the source-biased probing outperforms the *No Bias* prober and the *Query Bias 1* prober, resulting in an average source similarity that is initially 35% higher down to 13% after 100 documents have been extracted. Similarly, the source-biased prober outperforms the *Query Bias 2* prober, resulting in an average source similarity that is initially 57% higher down to 18% after 100 documents. Clearly, the higher focus value means the higher success for a probing algorithm.

Figure 5 shows another experiment where we also identified, in our set of 100 source–target pairs, all of those pairs that were a priori similar (e.g., `comp.sys.mac.apps` and `comp.sys.mac.system`) or dissimilar (e.g., `rec.crafts.textiles.sewing` and `comp.lang.perl.misc`). We show the relative performance of the *Source Bias, Query Bias 1*, and *No Bias* probers against these similar and dissimilar pairs. The *Query Bias 2* results track closely with the *Query Bias 1* results, and we drop them from this figure. The source-biased prober requires fewer documents to

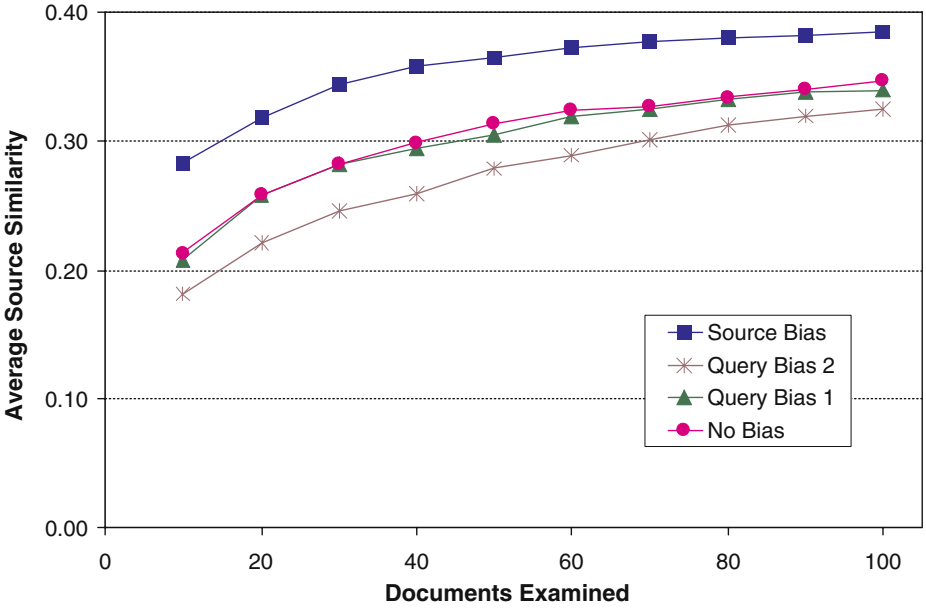

Figure 4 Probing efficiency for 100 source–target pairs.

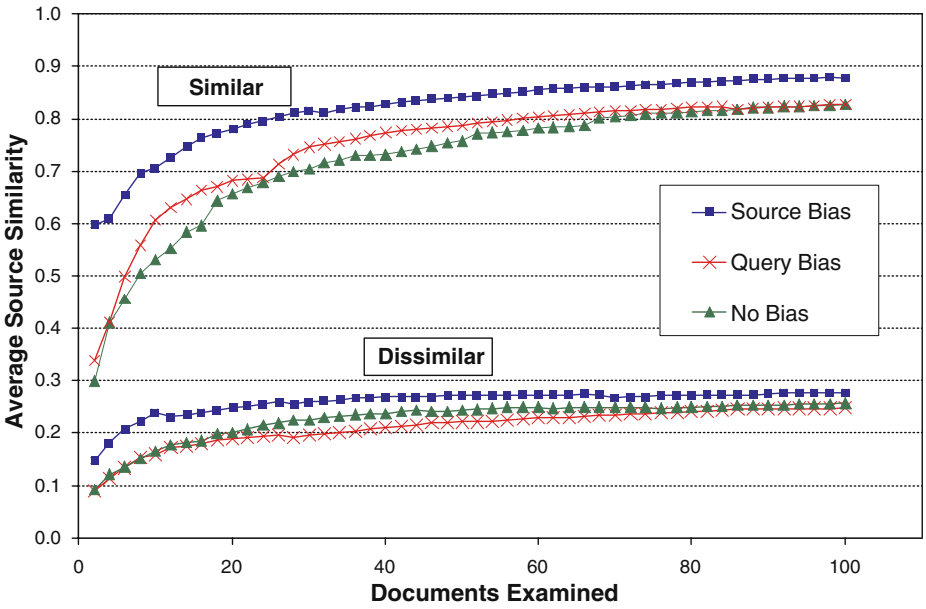

Figure 5 Probing efficiency for similar and dissimilar pairs.

achieve the same relevance level as the other probers for all 100 source–target pairs and for the similar and dissimilar pairs. For example, for the similar source–target pairs in Figure 5, the source-biased prober identifies target documents with 0.8 focus after extracting only 30 documents. In contrast, the other probers require between two and three times as many documents to achieve the same quality.

The third experiment is shown in Figure 6. Here we want to show how quickly a source-biased prober can hone on the most source-relevant documents in a target by plotting the percentage of the documents extracted that are similar (relevant) to the source for each of the four probers. As shown in Figure 6, the source-biased prober performs nearly two-times better than other probers: over 70% of the first 10 documents extracted from a target are source-relevant, whereas the other probers identify between 25 and 45% source-relevant documents. As more documents are examined for each target, the source-biased prober continues to maintain an advantage over the other probers. Since the source-biased prober extracts the highest-quality source-related documents from the target database first, we see the gradual decline in the *Source Bias* line, meaning that it performs the best in extracting relevant documents. We see fluctuations in the other approaches due to the randomness inherent in the query selection process. Unlike the source-biased prober which sends its best queries first, the other query probers may send a high-quality (source-relevant) query at any point of the querying process (or not at all), leading to the fluctuations in the quality of the extracted documents.

7.2 Ranking effectiveness with biased focus

The second set of experiments intends to evaluate how well source-biased probing compares with the alternative techniques when it comes to evaluating a collection of

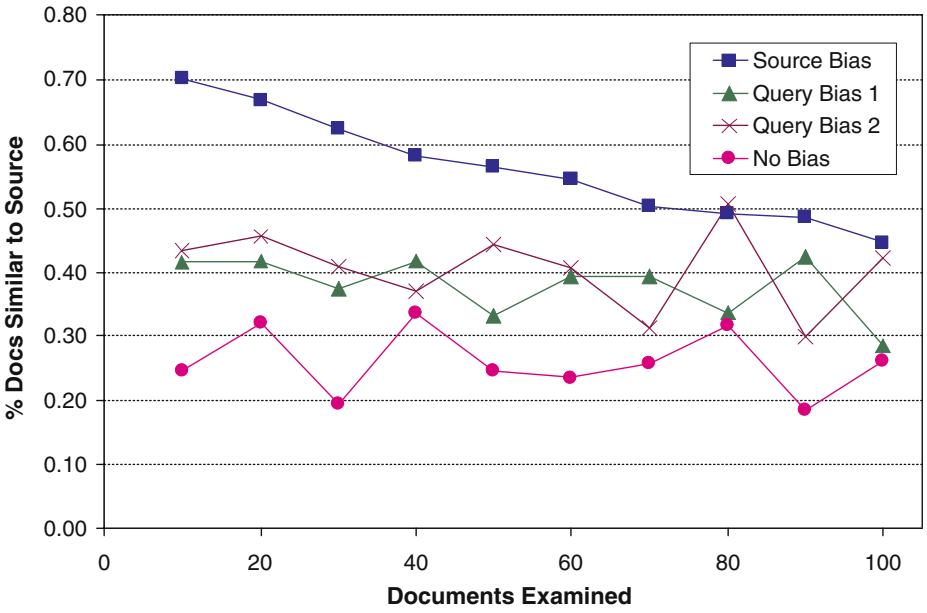

Figure 6 Average document quality for 100 pairs.

target databases. We performed two experiments using the deep web collection. In the first, we use PubMed as the source and examine all 50 deep web databases as targets. We computed the biased focus score using $Cosine\_focus_{\sigma}(\tau)$ and then ranked all target databases relative to PubMed using the biased focus measure. Since the deep web sites do not support random document selection, we are unable to evaluate an unbiased prober. So this experiment only compares the source-biased prober with query biased prober 1. Table 2 shows the top-10 ranked sites relative to PubMed. In the *Source Bias* column we also list in parenthesis the rank of each site assigned by the *Query Bias* prober.

The query-biased prober identifies several health-related sites in the deep web collection, but it mistakenly lists Linux Journal ahead of HealthAtoZ, as well as listing a web development site (DevGuru) and a genealogical magazine (FamilyTree) ahead of the health-related Mayo Clinic. Overall, only four of the top-ten sites

Table 2 Identifying databases relevant to PubMed.

Query bias	Source bias
1. AMA	1. Open Directory (13)
2. WebMD	2. Google (27)
3. Linux Journal	3. About (11)
4. HealthAtoZ	4. WebMD (2)
5. DevGuru	5. AMA (1)
6. FamilyTree Magazine	6. HealthAtoZ (4)
7. Mayo Clinic	7. Monster (22)
8. Novell Support	8. Mayo Clinic (7)
9. Random House	9. Random House (9)
10. January Magazine	10. BBC News (12)

could be considered topically relevant to PubMed. In contrast, the source-biased prober's top-eight sites are somewhat relevant to PubMed. In addition to the health-related sites, the source-biased prober also identifies three general sites that offer access to medical literature (Open Directory, Google, and About) that are ranked significantly lower by the query-biased prober. Interestingly, the source-biased prober identifies a fair number of scientific and bioinformatics-related job descriptions in the Monster jobs site, resulting in its high relevance (similarity) score to PubMed (high biased focus value). While we are encouraged by the rankings here, we note that there are some problems—for example, the three general sites (Open Directory, Google, and About) are ranked ahead of the three more medically-related sites (WebMD, AMA, and HealthAtoZ). In the next section, we show how the bilateral assessment of focus for identifying relationship sets may overcome these problems.

In the second experiment, we use Google as the source and examine all 50 deep web databases as targets, using the setup as described above. Table 3 shows the top-10 ranked sites relative to Google. In the *Source Bias* column we also list in parenthesis the rank of each site assigned by the *Query Bias* prober.

The query-biased prober identifies only one meaningful related site (About) in the top-10, whereas the Google source-biased prober finds both relevant sites in the top-10—About and the Open Directory Project—and ranks them as the top-2 most relevant sites. These results are further confirmation of the impact of the source-biased approach.

As a further illustration, for the source Linux Journal, we found that the top 10 source-biased probes are: linux, web, system, software, kernel, time, source, journal, user, file. These probes are representative of the coverage of Linux Journal and are surely more effective for discovering target databases relevant to Linux Journal than random probes. On inspection, we found that when probing the jobs site Monster, the resulting Linux-biased Monster summary is skewed towards linux-related jobs and other technical jobs. In contrast, the unbiased Monster summary is less relevant to the Linux Journal since on average, Monster contains many other types of jobs besides those that are linux related.

To validate the quality of source-biased database evaluation, we next randomly selected 10 sources from the newsgroup collection to evaluate against the entire set of 780 newsgroups. We compared the three probers *Source Bias, Query Bias 1*, and *No Bias*. For each of the 10 sources, we measured relevance precision as the percentage of the top-10 ranked target databases that are considered relevant to the

Table 3 Identifying databases relevant to Google.

Query bias	Source bias
1. Metropolis Magazine	1. About (3)
2. More Business	2. Open Directory Project (20)
3. About	3. Webmonkey (7)
4. Linux Journal	4. Monster (6)
5. Family Tree Magazine	5. Metropolis Magazine (1)
6. Monster	6. Random House (12)
7. Webmonkey	7. Linux Journal (4)
8. DevGuru	8. Family Tree Magazine (5)
9. US Customs	9. HealthAtoZ (15)
10. January Magazine	10. January Magazine (10)

source using $Cosine\_focus_{\sigma}(\tau)$. Relevance judgments were determined by the consensus opinion of three volunteers. Note that we do not measure recall since it is so expensive to calculate, requiring a relevance judgment for each source versus *every* database in the newsgroup collection.

Table 4 shows the precision for the three probers after extracting 40 documents per target database. *Source Bias* results in the highest precision in seven of ten cases, tying with the next best prober in two cases, and losing outright in one case. For the lone failure, *Source Bias* does succeed after extracting 80 documents, indicating that the mistake may be attributable to the error inherent in probing very few documents. In general, the average precision of the source-biased prober is nearly double that of the next best prober.

In Figure 7 we show the average precision for the ten sources when increasingly more documents are extracted per target. The source-biased approach displays higher precision than both the query-biased and unbiased probers in all cases considered, especially when based on very few documents. We again note that the source-biased prober extracts higher quality (source-relevant) documents at the beginning of the probing, which is vitally important for limiting the amount of probing necessary to yield adequate comparisons, especially given the size and growth rate of the deep web. As we mentioned before, the stop probing condition is especially critical since eventually the source-biased target summary may converge to the unbiased summary as more and more documents are extracted. In this case, we can see that as the number of documents extracted increases, the two alternatives to source-biased probing improve, but still lag significantly.

While we are encouraged by the results in this section, we see that there is still some room for improvement. In the next section, we show how the bilateral assessment of focus for identifying relationship sets may yield even stronger results.

7.3 Identifying inter-database relationships

The third set of experiments is designed to evaluate the effectiveness of using the source-biased framework to support the identification of interesting inter-database relationships that the alternative schemes do not. As discussed in Section 4.3, the source-biased framework can identify both similarity-based relationship sets and hierarchical relationship sets for a pair of deep web databases or for a source database and a collection of target databases. Unlike the query-biased and unbiased

Table 4 Relevance precision for 10 source newsgroups.

Source	No bias	Query bias	Source bias
comp.unix.misc	0.1	0.0	0.7
gnu.emacs.help	0.1	0.3	0.4
rec.aviation.owning	0.1	0.2	0.4
rec.games.chess.misc	0.1	0.1	0.6
rec.org.sca	0.1	0.0	0.4
sci.physics.research	0.5	0.3	0.8
talk.religion.misc	0.1	0.1	0.6
soc.culture.hawaii	0.2	0.1	0.2
rec.pets.cats.misc	0.1	0.1	0.1
comp.sys.mac.system	0.4	0.0	0.1

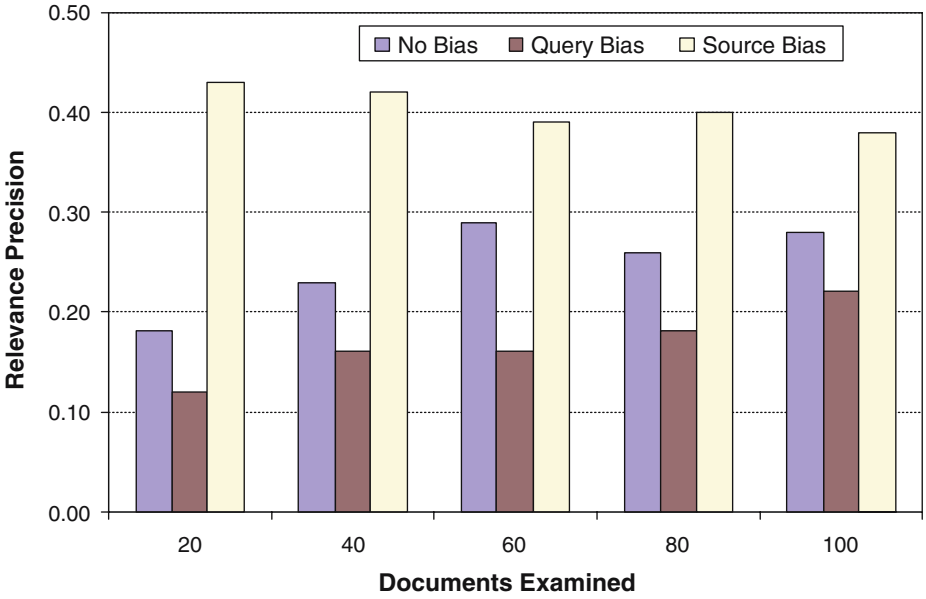

Figure 7 Average relevance precision.

probers, the asymmetric nature of source-biased probing allows us to characterize the nature of the relationship beyond the single relevance ranking using biased focus measure. Identifying relationship sets requires that each database be probed once for each source of bias considered, meaning that it can be more expensive than simple query-based probing.

We first illustrate relationship sets for PubMed over the deep web collection. In Table 5 we show four classes of relationship sets for $\lambda_{high} = 0.15$, $\lambda_{low} = 0.05$, and $\lambda_{diff} = 0.10$ using the source-biased prober described above. In contrast to the simple relevance ranking in Table 2, we see how the source-biased framework can differentiate between the very similar resources (the λ-equivalent sites WebMD, AMA, and HealthAtoZ) and the more general resources (the λ-superset sites Open Directory, Google, and About) relative to PubMed. In addition, we can identify sites with some common content (the λ-overlap sites Monster and Mayo Clinic) and sites concerned with significantly different topics (the λ-mutex sites Silicon Investor, Usenet Recipes, and Film Critic).

Similarly, we show in Table 6 several interesting relationships derived from the newsgroup collection for $\lambda_{high} = 0.70$, $\lambda_{low} = 0.40$, and $\lambda_{diff} = 0.30$ using the *Source Bias* prober discussed before. For each source considered, we probed all databases in the newsgroup collection, evaluated the biased focus metric, and report in Table 6 the relationships discovered. Again, by relying on the source-biased database analysis we may characterize relationships sets for each source that are helpful for answering relationship-centric queries of the kind posed at the beginning of the paper.

As an example, we identify `sci.physics.particle` as a member of the λ-subset relationship set of the mixed topic newsgroup `mixed11`, which consists of 25% physics-related articles in addition to articles on backgammon, juggling, and

Table 5 Source-biased analysis: identifying relationships relative to PubMed.

Resource (D)	URL	Description	$focus_{PM}(D)$	$focus_{D}(PM)$	Relationship
WebMD	http://www.webmd.com	Health/ Medical	0.23	0.18	λ-equivalent
AMA	http://www.ama-assn.org	Health/ Medical	0.19	0.16	λ-equivalent
HealthAtoZ	http:// www.healthatoz.com	Health/ Medical	0.18	0.16	λ-equivalent
Open Directory	dmoz.org	Web Directory	0.44	0.08	λ-superset
Google	http://www.google.com	Web Search Engine	0.37	0.10	λ-superset
About	http://www.about.com	Web Channels	0.25	0.08	λ-superset
Monster	http://www.monster.com	Jobs	0.14	0.08	λ-overlap
Mayo Clinic	http:// www.mayoclinic.com	Health/ Medical	0.12	0.11	λ-overlap
Silicon Investor	http:// www.siliconinvestor.com	Finance	0.03	0.04	λ-mutex
Usenet Recipes	http:// www.recipes2.alastra.com	Recipes	0.02	0.03	λ-mutex
Film Critic	http://www.filmcritic.com	Movies	0.01	0.03	λ-mutex

telecommunications. Interestingly, we can see that there are several overlapping relationships between newsgroups in related but slightly different fields (e.g., the two sports newsgroups `rec.sport.volleyball` and `rec.sport.cricket` and the game-related newsgroups `rec.games.go` and `rec.games.chess.misc`). Finally, we also identify several unrelated newsgroups, including `comp.sys.mac.system` relative to `misc.immigration.usa` and `comp.lang.c++` relative to `talk.religion.misc`.

Table 6 Source-biased analysis: identifying relationships in the newsgroup collection.

A	B	$focus_{A}(B)$	$focus_{B}(A)$	Relationship
comp.sys.mac.apps	comp.sys.mac.system	0.86	0.76	λ-equivalent
comp.sys.mac.system	comp.sys.mac.advocacy	0.79	0.74	λ-equivalent
sci.physics.particle	sci.physics	0.86	0.80	λ-equivalent
sci.physics.particle	mixed45	0.86	0.62	λ-subset/superset
comp.unix.misc	mixed120	0.91	0.56	λ-subset/superset
rec.boats.paddle	mixed11	0.88	0.57	λ-subset/superset
rec.sport.volleyball	rec.sport.cricket	0.47	0.46	λ-overlap
rec.games.go	rec.games.chess.misc	0.50	0.53	λ-overlap
comp.os.linux	comp.unix	0.43	0.48	λ-overlap
rec.crafts.textiles.sewing	comp.lang.perl.misc	0.35	0.32	λ-mutex
comp.sys.mac.system	misc.immigration.usa	0.23	0.36	λ-mutex
comp.lang.c++	talk.religion.misc	0.21	0.29	λ-mutex

7.4 Probing with focal terms

In our fourth set of experiments, we consider the impact of focal term probing on the success rate of source-biased probing. We evaluate four flavors of focal term probing—with the number of focal term groups k from which to draw source-biased probes set to 2, 3, 5, and 10. In our initial experiments with focal term probing, we discovered that there was little impact on either the efficiency of probing or the quality of target database evaluation when considering sources from the single-topic newsgroup collection.

In contrast, we discovered that focal term probing had a significant impact when used on mixed topic newsgroups, in which there are documents from several unrelated single topic newsgroups. In Figure 8, we show the probing efficiency for the four focal term source-biased probers relative to the best basic source-biased prober for 10 source–target pairs from the newsgroup collection. In each case, the sources were drawn exclusively from the mixed topic newsgroups.

All of the focal term techniques resulted in more efficient probing versus basic source-biased probing and only minor differences in ranking precision and relationship set generation quality, indicating that focal term probing can be advantageous in certain circumstances. Our intuition is that identifying focal terms is considerably more important in cases in which there are clear distinctions in term distributions as would be reflected in the mixed topic newsgroups in which several groups of documents are concerned with different topics.

7.5 Varying key parameters

For our final set of experiments, we evaluate the impact of several key parameters on the efficiency of source-biased probing. Again, we selected 10 sources and 10

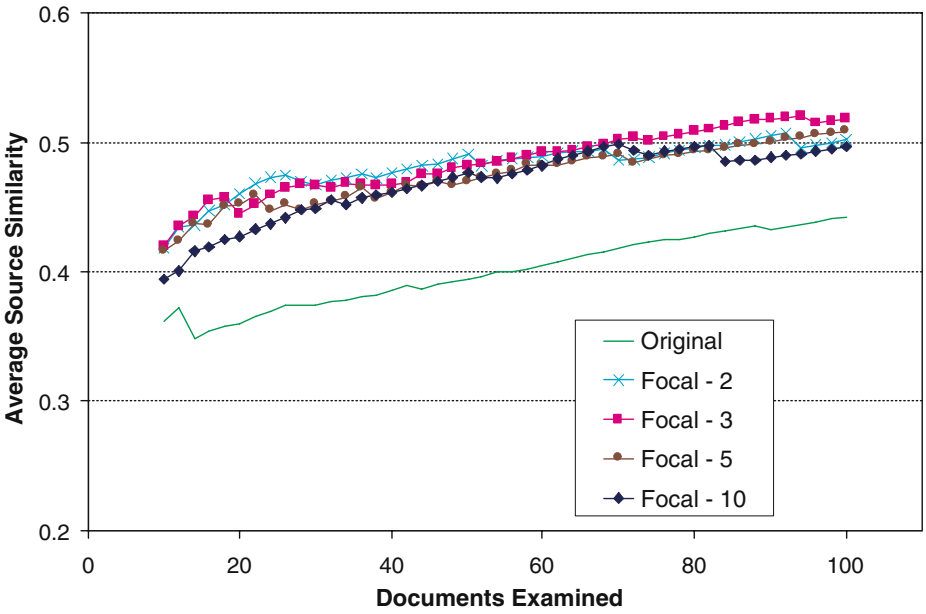

Figure 8 Impact of focal term probing.

targets at random from the entire newsgroup collection, resulting in 100 source–target pairs. The first parameter we consider is the choice of query selection for source-biased probing. We consider three alternatives: random probe selection (*Source Bias (random)*), probe selection based on the overall frequency of terms in the source summary (*Source Bias (dbFreq)*), and probe selection based on the document count of each term in the source summary (*Source Bias (docCount)*). We show the results in Figure 9. The two frequency-based measures result in approximately the same quality of extracted document, requiring the extraction of fewer documents to achieve the same relevance level as the randomized source-biased prober. Since the set of candidate probes in a source's resource summary is large (on the order of 1000 s), it seems reasonable to conclude that a random prober has a high likelihood of selecting non-discriminating probe terms, and hence extracting documents that are not relevant to the source of bias.

The second parameter we consider is the number of documents retrieved for each query. We considered the *Source Bias* prober described above, but we now vary the number of documents we retrieve per query, from 5 up to 20. As you can see in Figure 10, there is little change, so varying retrieved documents appears not to have a significant impact on the quality of source-biased probing.

The third parameter we compare is the choice of focus measure. In Figure 11, we compare the three versions of focus first discussed in Section 4.2: $Cosine\_focus_{\sigma}(\tau)$, $TW\ focus_{\sigma}(\tau)$, and $CT\ focus_{\sigma}(\tau)$. The dashed lines indicate the actual value of the overall focus measures calculated based on the actual resource summaries of the sources and targets. The upper dashed line corresponds to the actual $TW\ focus_{\sigma}(\tau)$. The other two dashed lines correspond to $Cosine\_focus_{\sigma}(\tau)$ and $CT\ focus_{\sigma}(\tau)$ and are overlapping. The first critical point to note is that both the $TW\ focus_{\sigma}(\tau)$ and $CT\ focus_{\sigma}(\tau)$ are slow to approach the actual overall focus as indicated by the dashed lines, indicating that substantially more documents must be extracted per target before the estimated focus can be considered reliable. In contrast, the cosine-based focus approaches the actual focus in only 45 documents on average, further bolstering our claims for its use. Additionally, we note that $Cosine\_focus_{\sigma}(\tau)$ slightly over-estimates the actual focus. This is reasonable, since for source-biased estimates based on very few documents, we would expect to identify high-quality documents first. Hence, the focus should be an overestimate. As the number of documents

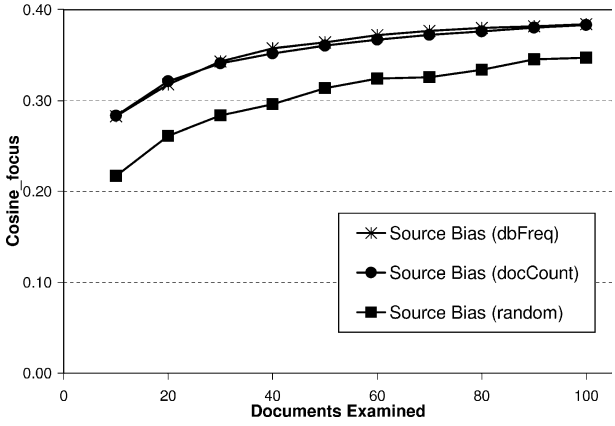

Figure 9 Query selection comparison.

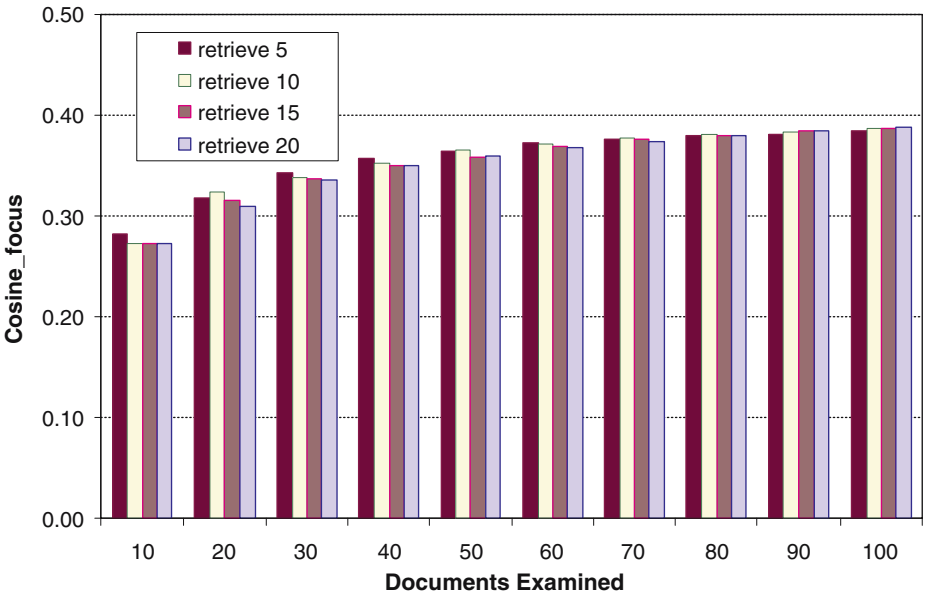

Figure 10 Documents retrieved comparison.

examined in a target nears the total available in the target, this upward bias should disappear.

One of the critical parameters to the overall success of source-biased probing with respect to comparing a source and a target is the quality of the original source

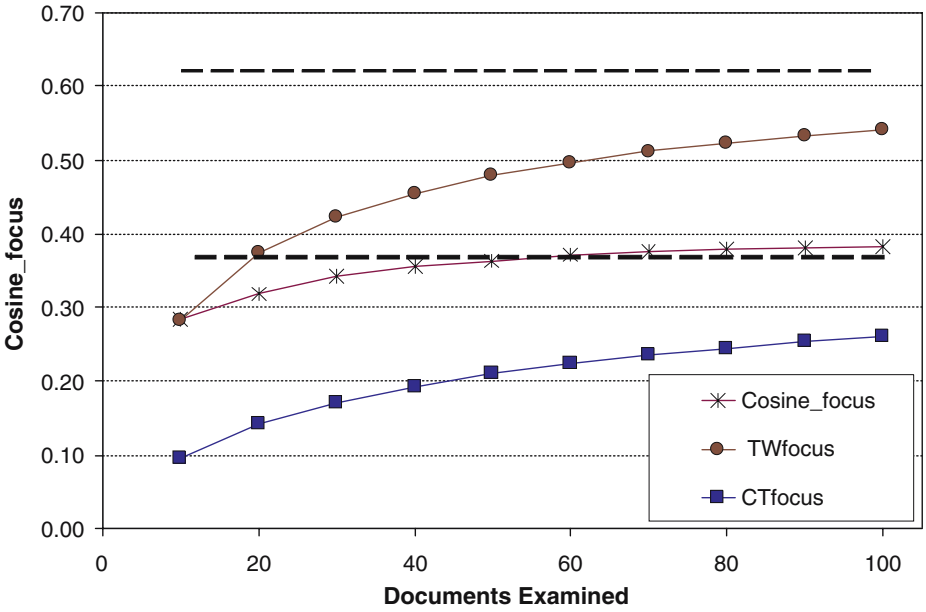

Figure 11 Comparison of three focus measures.

summary. In this set of experiments, we investigate the degree to which the source summary quality impacts the effectiveness of source-biased probing. We again randomly selected five sources to compare to the entire set of 780 groups. For each source D_i ($1 \leq i \leq 5$), we constructed three resource summaries by extracting either 100, 10, or 1% of the documents using the *No Bias* prober.

A resource summary based on 100% of the documents is exactly the actual summary ASummary(D_i). The resource summaries based on 10 and 1% of the total documents are estimated summaries of decreasing quality. For each of the five sources, we identified the number of relevant groups in the entire dataset (call this total r for each source). For non-obvious cases, we determined relevance by the consensus opinion of three volunteers. We then evaluated the summary quality by collecting 20 documents for each candidate target and ranking the targets by the $Cosine\_focus_{\sigma}(\tau)$ metric. We calculated the effectiveness for each source as the percentage of relevant targets ranked in the top-r. In Figure 12, we show the relevance ranking effectiveness for each source for each of the three resource summaries. The overall relevance is not impacted very much by the degradation in the quality of the resource summary. In three of the cases, the relevance either remains the same or falls slightly as the source summary quality decreases in quality. In two of the cases tested, the relevance precision increases slightly as the source summary quality decreases in quality. We attribute this phenomenon to the randomness inherent in the target summary probing, and aim to study it further in future work.

In our final experiment, we further illustrate how source summaries based on fairly small data samples may perform nearly as well as the actual source summaries for evaluating a target database. In Figure 13, we show the impact of the source summary quality for one source–target pair (including additional data for a 50% summary and a 5% summary). Interestingly, the relative slope for each curve is approximately the same, with only the 1% summary shifted down significantly from

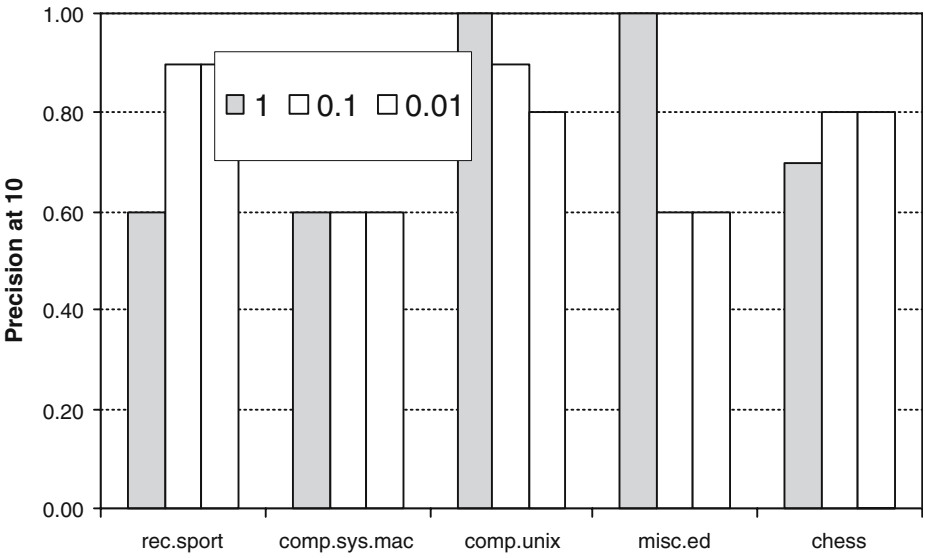

Figure 12 Impact of source summary quality on ranking.

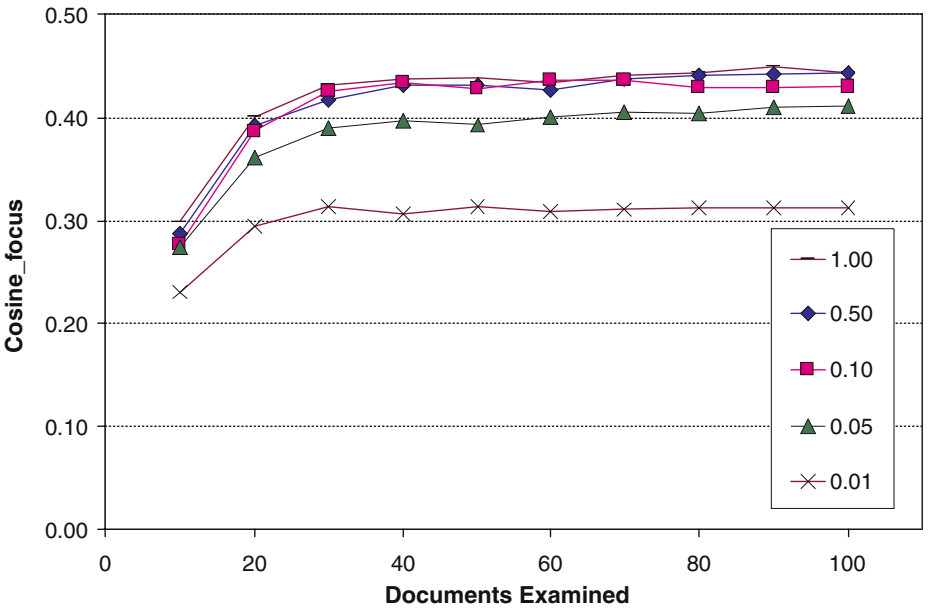

Figure 13 Impact of resource summary quality.

the performance of the actual (100%) summary. This suggests that relying on fairly small source samples (e.g., 5%) may be reasonable for evaluating deep web databases.

8 Conclusions and future work

We have presented a novel source-biased approach to efficiently discover interesting relationships among deep web databases. Our source-biased approach supports a relationship-centric view over a collection of deep web databases through source-biased probing and source-biased relevance metrics. Concretely, we have shown that the source-biased approach allows us to determine in very few interactions whether a target database is relevant to the source database by probing the target with very precise probes. The biased focus measure allows us to evaluate the relevance of deep web databases discovered and identify interesting types of source-biased relationships for a collection of deep web databases. Additionally, we have introduced source-biased probing with focal terms as a performance optimization to further improve the effectiveness of the basic source-biased model. Our experiments show that the source-biased approach outperforms query-biased probing and unbiased probing in most of the cases we have examined.

Our research on source-biased relevance reasoning continues along several dimensions. Because the source-biased probing framework relies on estimation at several key junctures, the introduction of errors in relevance evaluation and relationship identification must be carefully monitored. Error is introduced first through creation of the unbiased summary of the source which serves as the dictionary of candidate source-biased probes, then through the probing estimation

of the target, and finally through the reliance on term-based biased focus metrics for the evaluation of the underlying relevance of one deep web database to another. Measuring and minimizing this error may be important in many situations. On the theoretical side, our efforts are directed at extending the source-biased approach to a deep web database neighborhood graph for further analysis of relationships among multiple deep web databases. We are interested in extending the notion of relationship sets to account for inferences across deep web database pairs for which there may be no direct probing. For example, from Table 6, we observe that λ-*equivalent*(*comp.sys.mac.apps*) = {*comp.sys.mac.system*} and that λ-*equivalent* (*comp.sys.mac.system*) = {*comp.sys.mac.apps*, *comp.sys.mac.advocacy*}. Under what circumstances may we infer that *comp.sys.mac.advocacy* should also be in λ-*equivalent*(*comp.sys.mac.apps*), since an inference procedure is likely to break down as less direct evidence is available. Additionally, we are interested in comparing the clustering-based approach of the focal term probing algorithm with alternative grouping algorithms, like the popular database technique of frequent itemset mining. Finally, on the practical side we are continuing the development of the DynaBot system for crawling and analyzing deep web databases using the source-biased approach for discovering interesting relationships.

Acknowledgments This research is partially supported by the US National Science Foundation (NSF) CNS CCR, NSF ITR, US Department of Energy SciDAC, US Defense Advanced Research Projects Agency, CERCS Research Grant, IBM Faculty Award, IBM SUR grant, HP Equipment Grant, and LLNL LDRD.

References

1. Agichtein, E., Ipeirotis, P., Gravano, L.: Modeling query-based access to text databases. In: Proceedings of the International Workshop on the Web and Databases (WebDB '03), San Diego, 2003
2. Baeza-Yates, R.A., Ribeiro-Neto, B.A.: Modern Information Retrieval. ACM, Addison-Wesley (1999)
3. Bergman, M.: The deep web: Surfacing hidden value. BrightPlanet (2000)
4. Callan, J.P., Connell, M.E.: Query-based sampling of text databases. ACM Trans. Inf. Sys. **19**(2), 97–130 (2001)
5. Callan, J.P., Lu, Z., Croft, W.B.: Searching distributed collections with inference networks. In: Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '95), Seattle, 1995
6. Callan, J., Connell, M., Du, A.: Automatic discovery of language models for text databases. In: Proceedings of the 1999 ACM Conference on Management of Data (SIGMOD '99), Philadelphia, 1999
7. Caverlee, J., Liu, L., Buttler, D.: Probe, cluster, and discover: Focused extraction of qa-pagelets from the deep web. In: Proceedings of the 20th IEEE International Conference on Data Engineering (ICDE '04), Boston, 2004
8. Chakrabarti, S., van den Berg, M., Dom, B.: Focused crawling: A new approach to topic-specific Web resource discovery. In: Proceedings of the Eighth International World Wide Web Conference (WWW '99), May, 1999
9. Chang, K.C.-C., He, B., Li, C., Patel, M., Zhang, Z.: Structured databases on the web: Observations and implications. SIGMOD Rec. **33**(3) (2004)
10. Cohen, W.W., Singer, Y.: Learning to query the web. In: AAAI Workshop on Internet-Based Information Systems, 1996
11. Craswell, N., Bailey, P., Hawking, D.: Server selection on the World Wide Web. In: Proceedings of the Fifth ACM conference on Digital Libraries (ACM DL '00), San Antonio, 2000

12. Dolin, R., Agrawal, D., Abbadi, A.: Scalable collection summarization and selection. In: Proceedings of the Fourth ACM conference on Digital Libraries (ACM DL '99), Berkeley, 1999
13. French, J.C., Powell, A.L., Callan, J.P., Viles, C.L., Emmitt, T., Prey, K.J., Mou, Y.: Comparing the performance of database selection algorithms. In: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '99), Berkeley, 1999
14. Fuhr, N.: A decision-theoretic approach to database selection in networked IR. ACM Trans. Inf. Sys. **17**(3), 229–249 (1999)
15. Gravano, L., García-Molina, H.: Generalizing GlOSS to vector-space databases and broker hierarchies. In: Proceedings of the 21st International Conference on Very Large Databases (VLDB '95), Zurich, 1995
16. Gravano, L., García-Molina, H., Tomasic, A.: GlOSS: Text-source discovery over the Internet. ACM Trans. Database Syst. **24**(2), 229–264 (1999)
17. Hawking, D., Thistlewaite, P.: Methods for information server selection. ACM Trans. Inf. Sys. **17**(1), 40–76 (1999)
18. Ipeirotis, P.G., Gravano, L.: Distributed search over the hidden web: Hierarchical database sampling and selection. In: Proceedings of the 28th International Conference on Very Large Databases (VLDB '02), Hong Kong, 2002
19. Ipeirotis, P.G., Gravano, L., Sahami, M.: Probe, count, and classify: Categorizing hidden-web databases. In: Proceedings of the 2001 ACM Conference on Management of Data (SIGMOD '01), Santa Barbara, 2001
20. Ipeirotis, P.G., Gravano, L., Sahami, M.: QProber: A system for automatic classification of hidden-web databases. ACM Trans. Inf. Sys. (TOIS) **21**(1), 1–41 (2003)
21. Ipeirotis, P.G., Ntoulas, A., Cho, J., Gravano, L.: Modeling and managing content changes in text databases. In: Proceedings of the 21st IEEE International Conference on Data Engineering (ICDE '05), 2005
22. Liu, L.: Query routing in large-scale digital library systems. In: Proceedings of the 15th IEEE International Conference on Data Engineering (ICDE '99), Sydney, 1999
23. Lyman, P., Varian, H.R.: How much information. http://www.sims.berkeley.edu/how-much-info-2003 (2003)
24. Meng, W., Liu, K.-L., Yu, C.T., Wang, X., Chang, Y., Rishe, N.: Determining text databases to search in the internet. In: Proceedings of the 24th International Conference on Very Large Databases (VLDB '98), New York, 1998
25. Meng, W., Yu, C.T., Liu, K.-L.: Detection of heterogeneities in a multiple text database environment. In: Proceedings of the Fourth IFCIS International Conference on Cooperative Information Systems (CoopIS '99), Edinburgh, 1999
26. Nie, J.: An information retrieval model based on modal logic. Inf. Process. Manag. **25**(5), 477–497 (1989)
27. Porter, M.F.: An algorithm for suffix stripping. Program **14**(3), 130–137 (1980)
28. Powell, A.L., French, J.C., Callan, J.P., Connell, M.E., Viles, C.L.: The impact of database selection on distributed searching. In: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '00), Athens, 2000
29. ProFusion: http://www.profusion.com
30. PubMed: http://www.ncbi.nlm.nih.gov/PubMed/
31. Qiu, Y., Frei, H.-P.: Concept-based query expansion. In: Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '93), Pittsburgh, 1993
32. Raghavan, S., Garcia-Molina, H.: Crawling the hidden web. In: Proceedings of the 27th International Conference on Very Large Databases (VLDB '01), Rome, 2001
33. Rocco, D., Caverlee, J., Liu, L., Critchlow, T.: Exploiting the deep web with dynabot: matching, probing, and ranking. In: Poster Proceedings of the 14th International World Wide Web Conference (WWW '05), 2005
34. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. In: Readings in Information Retrieval. Morgan Kauffman, San Francisco, CA, 1997
35. Salton, G., Wong, A., Yang, C.: A vector space model for automatic indexing. CACM **18**(11), 613–620 (1971)
36. Schutze, H., Pedersen, J.O.: A cooccurrence-based thesaurus and two applications to information retrieval. Inf. Process. Manag. **33**(3), 307–318 (1997)
37. Sugiura, A., Etzioni, O.: Query routing for web search engines: Architecture and experiments. In: Proceedings of the Ninth International World Wide Web Conference (WWW '00), Amsterdam, 2000

38. Wang, J., Wen, J.-R., Lochovsky, F., Ma, W.-Y.: Instance-based schema matching for web databases by domain-specific query probing. In: Proceedings of the 30th International Conference on Very Large Databases (VLDB '04), Toronto, 2004
39. Wang, W., Meng, W., Yu, C.: Concept hierarchy based text database categorization in a metasearch engine environment. In: Proceedings of the First International Conference on Web Information Systems Engineering (WISE '00), Hong Kong, 2000
40. Wu, W., Yu, C.T., Doan, A., Meng, W.: An interactive clustering-based approach to integrating source query interfaces on the deep web. In: Proceedings of the 2004 ACM Conference on Management of Data (SIGMOD '04), Paris, 2004
41. Xu, J., Croft, W.B.: Query expansion using local and global document analysis. In: Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '96), Zurich, 1996
42. Yuwono, B., Lee, D.L.: Server ranking for distributed text retrieval systems on the internet. In: Database Systems for Advanced Applications (DASFAA '97), Melbourne, 1997
43. Zhang, Z., He, B., Chang, K.C.-C.: Understanding web query interfaces: Best-effort parsing with hidden syntax. In: Proceedings of the 2004 ACM Conference on Management of Data (SIGMOD '04), Paris, 2004