Discovering and Ranking Web Services with BASIL: A Personalized Approach with Biased Focus

James Caverlee, Ling Liu, and Daniel Rocco
Georgia Institute of Technology
College of Computing
Atlanta, GA 30332, U.S.A.
{caverlee, lingliu, rockdj}@cc.gatech.edu *

ABSTRACT

In this paper we present a personalized web service discovery and ranking technique for discovering and ranking relevant data-intensive web services. Our first prototype – called BASIL – supports a *personalized* view of data-intensive web services through source-biased focus. BASIL provides service discovery and ranking through source-biased probing and source-biased relevance metrics. Concretely, the BASIL approach has three unique features: (1) It is able to determine in very few interactions whether a target service is relevant to the given source service by probing the target with very precise probes; (2) It can evaluate and rank the relevant services discovered based on a set of source-biased relevance metrics; and (3) It can identify interesting types of relationships for each source service with respect to other discovered services, which can be used as value-added metadata for each service. We also introduce a performance optimization technique called source-biased probing with focal terms to further improve the effectiveness of the basic source-biased service discovery algorithm. The paper concludes with a set of initial experiments showing the effectiveness of the BASIL system.

Categories and Subject Descriptors: H.3.5 [Online Information Services]: Web-based services

General Terms: Algorithms, Experimentation

Keywords: data-intensive services, biased discovery, ranking

1. INTRODUCTION

Most web services today are web-enabled applications that can be accessed and invoked using a messaging system, typically relying on standards such as XML, WSDL, and SOAP [29]. Many companies have latched onto the web services mantra, including major software developers, business exchanges, eCommerce sites, and search engines [15, 9, 2, 1, 7]. A large and growing portion of the web services today can be categorized as *data-intensive web services*.

*This research is partially supported by NSF CNS CCR, NSF ITR, DoE SciDAC, DARPA, CERCS Research Grant, IBM Faculty Award, IBM SUR grant, HP Equipment Grant, and LLNL LDRD.

ICSOC'04, November 15–19, 2004, New York, New York, USA.

Data-intensive web services provide access to huge and growing data stores and support tools for searching, manipulating, and analyzing those data stores. For example, both Amazon [1] and Google [7] now provide XML- and SOAP-based web service interfaces to their underlying data repositories with support for advanced search operators over, collectively, billions of items. In the life sciences domain, many bioinformatics services are transitioning from human-in-the-loop web interfaces to the web services model [9], providing direct access to unprecedented amounts of raw data and specialized research tools to provide high-level analysis and search over these data services.

With the increasing visibility of web services and the Service-Oriented Computing paradigm [18], there is a growing need for efficient mechanisms for discovering and ranking services. Effective mechanisms for web service discovery and ranking are critical for organizations to take advantage of the tremendous opportunities offered by web services, to engage in business collaborations and service compositions, to identify potential service partners, and to understand service competitors and increase the competitive edge of their service offerings.

Current web service discovery techniques can be classified into two types: categorization-based discovery and personalized relevance-based discovery. The former discovers web services by clustering and categorizing a collection of web services into different groups based on certain common properties of the services. Most of the existing UDDI [28] registry-based service discovery methods are of this type. They typically discover relevant services by querying metadata maintained in the common registries (like the ones offered by Microsoft [16] and IBM [10]). A typical question is "Which bioinformatics web services offer BLAST capability" or "Which commercial services offer online auctions". The second type of discovery mechanisms uses personalized relevance reasoning and support questions such as "Which services offer the same type of content as NCBI", and "Find the top-ten web services that offer more coverage than the BLAST services at NCBI". These two types of service discovery techniques offer different focus and complementary capabilities. Consider the following examples:

• A bioinformatics researcher may be interested in finding all services similar to NCBI's BLAST service for searching DNA and protein sequence libraries [17]. Current service registries may provide pointers to other BLAST services, but they do not describe how these other sites relate specifically to NCBI's BLAST service. Which services provide the most similar coverage with respect to NCBI (e.g. of similar proteins or organisms)? Which services are complementary in their coverage (e.g. of other sequence libraries)? How best should the BLAST services be ranked relative to the NCBI service?

• A health science researcher familiar with the PubMed med-

ical literature service may be interested in discovering other related medical digital library services. Given his prior knowledge with PubMed, he may want to ask certain personalized (source-biased) discovery requests, which are not supported by the conventional service-registry-based discovery model. Examples include: Find and rank all PubMed-related medical literature sites. Which services have more general coverage than PubMed? Which medical literature services are more specialized than PubMed?

These examples highlight two fundamental differences between the categorization-based and the personalization-based discovery model: (1) Categorization of web services based on general descriptions maintained at the service registries are insufficient and inadequate when a user is interested in discovery based on a particular service (or a set of services) with which she has prior experience or knowledge (e.g. NCBI BLAST or PubMed); and (2) There is a need for service ranking metrics that capture the relative scope and coverage of the data offered with respect to a previously known service. Although these two types of discovery mechanisms are complementary, most existing proposals on web service discovery fall into the first type. Surprisingly, there are, to our knowledge, no effective means to provide such *personalized* and *biased* discovery and ranking support without relying on significant human intervention.

In this paper we present algorithms for discovering and ranking relevant data-intensive web services. Our first prototype – called BASIL[1] — supports a *personalized* view of web services through source-biased probing and source-biased relevance detection and ranking metrics. Concretely, our approach is capable of discovering and ranking web services by focusing on the nature and degree of the *data relevance* of the source service to others. Given a service like NCBI's BLAST – called the *source* - the BASIL source-biased probing technique leverages the summary information of the source to generate a series of biased probes to other services – called the *targets*. This source-biased probing allows us to determine whether a target service is relevant to the source by probing the target with very few focused probes. We introduce the *biased focus* metric to discover and rank highly relevant data services and measure relevance between services. Our initial results on both simulation and web experiments show that the BASIL system supports efficient discovery and ranking of data-intensive web services.

2. MODEL AND PROBLEM STATEMENT

We consider a universe of discourse $\mathcal{W}$ consisting of D data-intensive web services: $\mathcal{W} = \{S_1, S_2, \ldots, S_D\}$ where each service produces one or more XML documents in response to a particular service request. Hence, we describe each web service S_i as a set of M_i documents: $S_i = \{doc_1, doc_2, \cdots, doc_{M_i}\}$. For example, the documents corresponding to the NCBI BLAST service would consist of genetic sequences and documentation generated in response to a service requests. Similarly, the documents corresponding to PubMed would consist of the set of medical journal articles in the PubMed data repository.

There are N terms (t_1, t_2, ..., t_N) in the universe of discourse $\mathcal{W}$ – including both the tags and content of the XML documents – where common stopwords (like 'a', 'the', and so on) have been eliminated. Optionally, the set of N terms may be further refined by stemming [19] to remove prefixes and suffixes.

Adopting a vector-space model [22, 23] of the service data repository, we describe each service S_i as a vector consisting of the terms in the service along with a corresponding weight:

$$\textsc{Summary}(S_i) = \{(t_1, w_{i1}), (t_2, w_{i2}), \cdots, (t_N, w_{iN})\}$$

[1]**BASIL: B**i**A**sed **S**ervice d**I**scovery a**L**gorithm

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

We call the vector $\textsc{Summary}(S_i)$ a *service summary* for the data-intensive web service S_i. A service summary is a single aggregate vector that summarizes the overall distribution of terms in the set of documents produced by the service. In this first prototype of BASIL, we rely on a bag-of-words model that is indifferent to the structure inherent in the XML documents. As we demonstrate in the experiments section, this bag-of-words approach is quite powerful without the added burden of structural comparisons. We anticipate augmenting future versions of BASIL to incorporate structural components (to support schema matching, leverage existing ontologies, etc.).

To find $\textsc{Summary}(S_i)$, we must first represent each document doc_j ($1 \leq j \leq M$) 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 service summary in a number of ways. Initially, the weight for each term in the service summary may be based on the overall frequency of the term across all the documents in the service (called the *service frequency*, or *servFreq*): $w_{ik} = servFreq_{ik} = \sum_{j=1}^{M} freq_{jk}$. 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*).

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

2.1 Estimating Service Summaries

Ideally, we would have access to the complete set of documents belonging to a data-intensive web service. We call a service summary for S_i built on these documents an *actual service summary* or $\textsc{ASummary}(S_i)$. However, the enormous size of the underlying repositories for many data-intensive web services coupled with the non-trivial costs of collecting documents (through repeated service requests and individual document transfer) make it unreasonable to generate an actual service summary for every service available. As a result, previous researchers in the context of distributed databases have introduced several *probing* techniques for generating representative summaries based on small samples of a document-based collections [3, 4]. We call such a representative summary an *estimated service summary*, or $\textsc{ESummary}(S_i)$:

$$\textsc{ESummary}(S_i) = \{(t_1, w_{i1}), (t_2, w_{i2}), \cdots, (t_N, w_{iN})\}$$

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 service summary since only

a fraction of the total documents in a service will be examined. The goal of a prober is typically to find ESUMMARY(S_i) such that the relative distribution of terms closely matches the distribution of terms in ASUMMARY(S_i), even though only a fraction of the total service documents will be examined.

Current probing techniques for estimating service summaries aim at estimating the overall summary of the data served by a web service. We classify them into two categories: random sampling and query-based sampling.

Random Sampling – No Bias

If we had unfettered access to a data-intensive web service, we could randomly select terms from the service to generate the estimated service summary ESUMMARY(S_i). Barring that, we could randomly select documents with which to base the estimated service 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 services only provide a request-response mechanism for extracting documents.

Query-based Sampling – Query Bias

As a good approximation to unbiased probing, Callan et al. [3, 4] have introduced a query-based sampling technique for generating accurate estimates of document-based collections by examining only a fraction of the total documents. The Callan technique has been shown to provide accurate estimates using very few documents (e.g. several hundred). Adapting the Callan technique to the web services context requires repeatedly requesting documents from a service using a limited set of service requests. Since the documents extracted are not chosen randomly, but are biased by the service request mechanism through the ranking of returned documents and by providing incomplete access to the entire data service repository, 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 the service. In the rest of the paper, when we refer to an estimated service summary ESUMMARY(S_i), we mean one that has been produced by a query-biased prober.

2.2 Comparing Service Summaries

In order to determine the relevance of one service S_i to another service S_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 service summaries. A fairly simple and straightforward approach is based on a count of the number of common terms in the two services S_i and S_j:

$$rel(S_i, S_j) = \frac{|\text{ESUMMARY}(S_i) \cap \text{ESUMMARY}(S_j)|}{max(|\text{ESUMMARY}(S_j)|, |\text{ESUMMARY}(S_i)|)}$$

Two services with exactly the same terms represented in their estimated summaries will have $rel(S_i, S_j) = 1$, indicating the highest possible degree of relevance. Conversely, two services with no terms in common will have $rel(S_i, S_j) = 0$, indicating the lowest possible degree of relevance.

We now use an example to illustrate why the existing service summary estimation techniques are inadequate for effectively discovering relevant services, especially in terms of the data coverage of one (target) in the context of the other (source).

Example: *We collected fifty documents from the Google web service, the PubMed web service, and ESPN's search site, respectively, using a query-based sampling technique for service summary estimation. Using the service summaries constructed, we find that* $rel(Google, PubMed) = 0.05$ *and* $rel(ESPN, PubMed) = 0.06$. *In both cases the service summaries share very few terms in common and hence both Google and ESPN appear to be irrelevant with respect to PubMed, even though Google provides considerable health-related content. Based on these figures, we could incorrectly conclude that: (1) Google is irrelevant to PubMed; and (2) Relatively speaking, ESPN is more relevant to PubMed than Google.*

This example underlines two critical problems with current techniques for probing and comparing service summaries:

First, current service summary estimation techniques are concerned with generating *overall* (or *global*) summaries of the underlying data repositories. The goal is to generate essentially an unbiased estimate of the actual service summary. Second, the current relevance comparison metric fails to serve as a valuable ranking metric or indicator of interesting relationships between target services in terms of the data coverage of a target service with respect to the source.

3. THE BASIL SYSTEM

Bearing these issues in mind, we now introduce BASIL – an efficient web service discovery and ranking prototype that relies on a *biased* perspective of services rather than on a single *global* perspective. BASIL relies on three fundamental steps: (1) source-biased probing for web service discovery; (2) evaluation and ranking of discovered services with the biased focus metric; and (3) leveraging the biased perspective of service sources and targets to discover interesting relationships.

3.1 Source-Biased Probing

Given a data-intensive web service – the *source* – the source-biased probing technique leverages the summary information of the source to generate a series of biased probes for analyzing another service – the *target*. This source-biased probing allows us to determine in very few interactions whether a target service is relevant to the source by probing the target with focused probes.

To help differentiate the source-biased approach from others discussed in Section 2, in this section we use σ to denote the source service and τ to denote the target service instead of S_i and S_j. Given two services σ and τ, the output of the source-biased probing is a subjective service summary for τ that is biased towards σ. We define the source-biased summary of the target service, denoted by $\text{ESUMMARY}_\sigma(\tau)$, as follows:

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

N is the total number of terms used in analyzing the set of data-intensive web services. w_i^σ $(1 \leq i \leq N)$ is the weight of term t_i, defined using one of the weight function introduced in Section 2. To distinguish the term weight w_j from the corresponding term weight in the biased target summary, we denote the bias by w_j^σ. 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 uses the estimated service summary of the source σ, denoted by ESUMMARY(σ), as a dictionary of candidate probe terms and sends a series of query requests parameterized by probe terms, selected from ESUMMARY (σ), to the target service τ; for each probe term, it retrieves the top m matched documents from τ, generates summary terms and updates $\text{ESUMMARY}_\sigma(\tau)$. This process repeats until a stopping condition is met. Figure 1 illustrates the source-biased probing process. Note that in this first prototype of BASIL the service requests are constrained to keyword-based probes. Note that the source-biased approach can also be applied to UDDI-directory-based discovery by restricting

SourceBiasedProbing(Source σ, Target τ)
For target service 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 service 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

the source summary to be generated from the meta-data description maintained at the registries rather than performing the source-biased probing directly. However, the quality of the discovery results will be lower due to the lack of richness in the metadata maintained at the service registries for many services.

Now we use a simple example to illustrate the power of source-biased probing. For presentation brevity, we are considering a simplistic world of only very few terms per service summary. In reality, each service summary would consist of orders of magnitude more terms:

Example: *Suppose that our goal is to understand the relevance of Google to PubMed. Suppose,* $\text{ESUMMARY}(PubMed) = \{arthritis, bacteria, cancer\}$ *(where for simplicity we have dropped the term weights from the summary). Again for simplicity suppose that Google provides access to only three types of information: health, animals, and cars:* $\text{ASUMMARY}(Google) = \{arthritis, bacteria, cancer, dog, elephant, frog, garage, helmet, indycar\}$. *An unbiased prober could result in* $\text{ESUMMARY}(Google) = \{arthritis, frog, helmet\}$, *whereas a source-biased prober could result in* $\text{ESUMMARY}_{PubMed}(Google) = \{arthritis, bacteria, cancer\}$. *This simple example illustrates the essence of the source-biased probing and how it accentuates the commonality between the two services.*

The performance and effectiveness of the source-biased probing algorithm depends upon a number of factors, including the selection criterion used for choosing source-specific candidate probe terms, and the type of stop condition used to terminate the probing process.

Mechanisms to Select Probe Terms

There are several possible ways to select the probes based on the statistics stored with each service 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 N_σ of all non-zero weighted terms in the unbiased source summary $\text{ESUMMARY}(\sigma)$.

Uniform Random Selection: In this simplest of selection techniques, each term that occurs in $\text{ESUMMARY}(\sigma)$ 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 service summary. For example, all terms in $\text{ESUMMARY}(\sigma)$ 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. *servFreq*, *docCount*, etc.), several flavors of weight-based selection may be considered.

Different Types of Stop Probing Conditions

The stop probing condition is the second critical component in 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 agnostic to the number of documents that are examined for each service.

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 service. Since some queries may return no documents, this stopping condition will require more query probes than the first alternative when $MaxProbes = MaxDocs$.

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 $\text{ESUMMARY}(\sigma)$. 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 $\text{ESUMMARY}_\sigma(\tau)$ 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.

Due to the space limitation, we refer readers to our technical report [5] for detailed experiments on the impact of these two parameters.

3.2 Evaluating and Ranking Services

Given a source and a target service, once we generate the source-biased summary for the target service, we need an efficient mechanism to evaluate the source-biased relevance of a target service with respect to the source. Once a set of target services have been evaluated with the source-biased relevance metric, we can then rank the target services with respect to the source of bias. We begin by discussing the necessary components of the source-biased metric.

Let σ denote a source service modeled by an estimated summary and τ denote a target service 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 service 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 data-intensive web services σ and τ with the focus in terms of σ by $focus_\sigma(\tau)$ or in terms of τ by $focus_\tau(\sigma)$.

We propose to use the well-known cosine similarity (or normalized inner product) to approximate the source-biased focus measure. We define the *cosine-based focus* as follows:

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

where $w_{\sigma k}$ is the weight for term k in $\text{ESUMMARY}(\sigma)$ and $w_{\tau k}^{\sigma}$ is the σ-biased weight for term k in $\text{ESUMMARY}_\sigma(\tau)$. 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. Intuitively, the cosine-based biased focus is appealing since it reasonably captures the relevance between two data-intensive web services.

Ranking Relevant Services

Given the biased focus measure, we may probe a group of tar-

get services to identify the most relevant services to the source of bias. For a single source of bias S_1 from our universe of discourse $\mathcal{W}$, we may evaluate multiple target services S_2, S_3, ..., S_d. For each target service, we may evaluate the appropriate focus measure for each source-target pair (i.e. $focus_{S_1}(S_2)$, $focus_{S_1}(S_3)$, etc.). We may then rank the target services in descending order in terms of their source-biased focus with respect to S_1.

As we will show in our experiments section, source-biased probing results in the identification of relevant services that existing approaches may overlook. We also show that source-biased probing can generate source-biased summaries of good quality using far fewer documents than existing approaches, placing significantly less burden on the target services.

3.3 Identifying Interesting Relationships

The critical third component of the BASIL system consists of the techniques for exploiting and understanding interesting relationships between services using a source-biased lens. By analyzing the nature of the relationships between data-intensive web services, we will provide support for understanding the relative scope and coverage of one service with respect to another.

The source-biased probing framework and biased focus measure provide the flexible building blocks for automated identification of interesting relationships between services, especially since the framework promotes an asymmetric source-biased view for any two services. Our relationship discovery module creates a flexible organization of services, where each service is annotated with a list of relationship sets. The two typical relationship types we have identified are similarity-based and hierarchical-based.

Similarity-Based Relationships

Given the universe of discourse $\mathcal{W} = \{S_1, S_2, \ldots, S_D\}$, we identify three similarity-based relationship sets for a particular service S_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 $focus_{S_i}(S_j) > \lambda_{high}$ and $focus_{S_j}(S_i) > \lambda_{high}$ hold, then we may conclude that S_i is sufficiently focused on S_j and S_j is sufficiently focused on S_i. Hence, the two services are approximately the same in terms of their data 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, $\lambda\text{-}equivalent(S_i) = \{\forall S_j \in \mathcal{W} \mid focus_{S_i}(S_j) > \lambda_{high} \wedge focus_{S_j}(S_i) > \lambda_{high}\}$.

$\lambda -$ **complement**: If both $focus_{S_i}(S_j) < \lambda_{low}$ and $focus_{S_j}(S_i) < \lambda_{low}$ hold, then we can conclude that S_i and S_j are sufficiently concerned with different topics since neither one is very focused on the other. We annotate this approximate complementary nature with the λ prefix. Formally, $\lambda\text{-}complement(S_i) = \{\forall S_j \in \mathcal{W} \mid focus_{S_i}(S_j) < \lambda_{low} \wedge focus_{S_j}(S_i) < \lambda_{low}\}$.

$\lambda -$ **overlap**: When two services S_i and S_j are neither λ-equivalent nor λ-complementary, we say that the two services λ-overlap. Formally, $\lambda\text{-}overlap(S_i) = \{\forall S_j \in \mathcal{W} \mid S_j \notin \lambda\text{-}complement(S_i) \wedge S_j \notin \lambda\text{-}equivalent(S_i)\}$.

Hierarchical Relationships

In addition to similarity-based relationship sets, we also define hierarchical relationship sets by measuring the relative coverage of target services in $\mathcal{W}$ with respect to a particular text service S_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 $focus_{S_i}(S_j) - focus_{S_j}(S_i) > \lambda_{diff}$, then a relatively significant portion of S_i is contained in S_j, indicating that S_j has a λ-superset relationship with S_j. We use the λ prefix to indicate that S_j is not a strict superset of S_i, but rather that the relationship is parameterized by λ_{diff}. Formally, $\lambda\text{-}superset(S_i) = \{\forall S_j \in \mathcal{W} \mid focus_{S_i}(S_j) - focus_{S_j}(S_i) > \lambda_{diff}\}$.

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

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

Using Relationship Sets Both similarity based and hierarchy-based inter-service relationships can be generated automatically, and used as metadata annotation to each of the services. These source-biased relevance data provide a flexible foundation for relationship analysis among services. For any service S_i, we need only consult the appropriate relationship set. The three similarity-based relationship sets provide the basis for answering queries of the form: "What other services 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 services are more general than X? Or more specialized than X?".

In addition, these relationship sets are useful for routing service requests to the appropriate services. For example, a user interested in BLAST data may choose to use both NCBI's BLAST service and all of the services that have a λ-*equivalence* relationship with NCBI BLAST. Alternatively, a user interested in maximizing coverage of multiple topically-distinct services, may choose to query both the source service she knows about and any members in the complementary set of the source service. The hierarchical relationship sets are particularly helpful in cases where a user may refine a service request to more specialized services, or alternatively, may choose to generalize the scope of the service request by considering services further up the hierarchy to get more matching answers.

4. FOCAL TERM PROBING

One of the critical parameters to the success of BASIL's 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 introduce 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*. A unique advantage of using focal terms is that the biased summaries of target services can be generated in fewer queries with higher quality.

4.1 Focal Terms and Focal Term Groups

Let σ denote a source service with its unbiased service 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 service 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

Table 1: Example Focal Terms for PubMed

1	care, education, family, management, ...
2	brain, gene, protein, nucleotide, ...
3	clinical, noteworthy, taxonomy, ...
4	experimental, molecular, therapy, ...
5	aids, evidence, research, winter, ...

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.

Formally, we need an algorithm that can find a partition of $Terms(\sigma)$ into k focal term groups: $Terms(\sigma) = \{FTerms_1, \ldots, FTerms_i, \ldots, FTerms_k | \bigcup_{i=1}^{k} FTerms_i = \{t_1, ..., t_n\}$ 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.

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 service, 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 those generated by 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.

It is important to note that, unlike previous research in grouping terms – for query-expansion [31, 21] or finding similar terms [24] – 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 service 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.

4.2 Finding 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 2, we view a service S_i as a set of documents, each of which is described by a vector of terms and weights. We now invert our view of a service using the same set of information. We consider a service S_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: $Terms(S_i) = \{term_1, term_2, \cdots, term_N\}$. For the N terms in the service, 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 $Terms(S_i)$ into k clusters. Given the term vectors and the similarity function, a number of clustering algorithms can be applied to partition the set $Terms(S_i)$ 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 $Terms(S_i)$ 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:

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

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 service 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:

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

In Section 5 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.

FocalTerms(Num 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 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 $\delta_1, \delta_2, \ldots, \delta_k$
      mu_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

4.3 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 service. We propose a simple round-robin selection technique whereby a single term is selected from each focal term group in turn. 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.

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. For each focal term group, we may decide which term to select for each cycle by using one of the selection criteria discussed in Section 3.

5. EXPERIMENTS

In this section, we describe four sets of experiments designed to evaluate the benefits and costs of BASIL. The first set intends to show the effectiveness of our source-biased probing algorithm and compare its performance with query-biased probing and unbiased probing. The second set evaluates the biased focus measure as an effective tool for ranking services. The third set shows the efficiency of the biased focus measure in identifying interesting inter-service relationships. The fourth set 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 focal term probing can achieve about ten percent performance improvement over the basic algorithm for source-biased probing.

Since there are no large data-intensive web service collections for experimentation, we rely on: (1) a large collection of newsgroups designed to emulate the diversity and scope of real-world data-intensive web services; and (2) a modest collection of real-world web sources. Since the services in the 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.

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, 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 services, 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 2.5GB worth of articles in 780 groups.

Web Collection: For the second collection, we randomly selected 50 sites from the ProFusion [20] directory of web sites that support queries, 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.

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 estimated service summary based on the overall term frequency of each term ($servFreq$). 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).

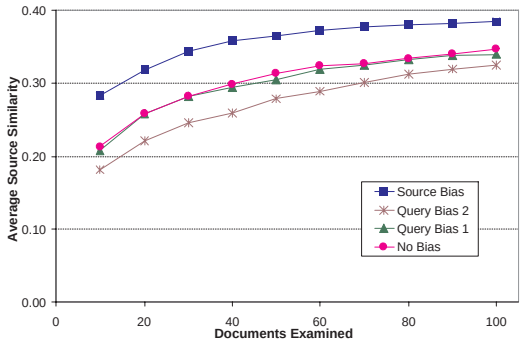

Figure 3: Probing Efficiency for 100 Pairs

5.1 Effectiveness of Source-Biased Probing

The goal of our first set of experiments is to compare source-biased probing with existing probing techniques and to evaluate the efficiency and quality of source-biased probing. The source-biased probing show significant gain in terms of the percentage of documents probed that are similar to the source.

We first 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 $servFreq$; 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 $servFreq$ of the target's service 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 5 documents per probe query from each target.

In Figure 3, 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 by about 10% and outperforms the *Query Bias 2* prober by about 15%. Clearly, the higher focus value means the higher success for a probing algorithm.

Figure 4 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. mac.apps and mac.system) or dissimilar (e.g. textiles.sewing and perl.misc). We show the relative performance of the *Source Bias*, *Query Bias 1*, and *No Bias*

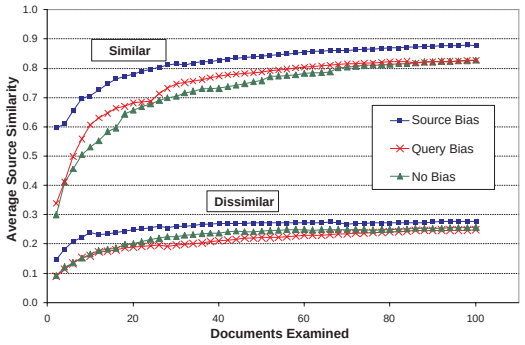

Figure 4: Probing Efficiency Breakdown

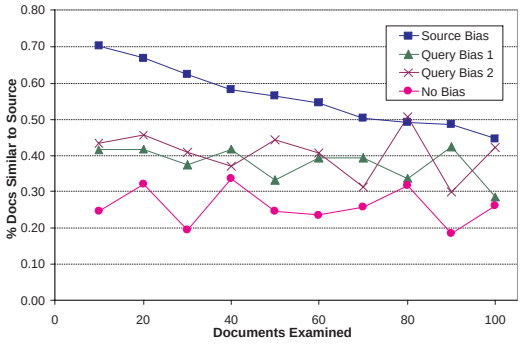

Figure 5: Average Document Quality for 100 Pairs

probers against these similar and dissimilar pairs. The source-biased prober requires fewer documents to 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 4, the source-biased prober identifies target documents with 0.8 focus after extracting fewer than 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 5. 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 5, 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.

5.2 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 and ranking collection of target services. We use PubMed as the source and examine all 50 web sites as targets. We computed the biased focus score using $Cosine\_focus_{\sigma}(\tau)$ and then ranked all targets relative to PubMed using the biased focus measure. Since the 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

Table 2: Identifying Web Sources 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)

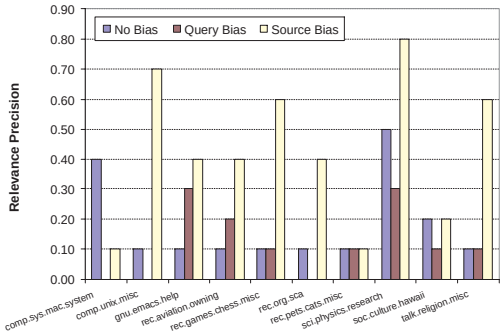

Figure 6: Precision for 10 Source Newsgroups

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 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 could be considered topically relevant to PubMed. In contrast, the source-biased prober's top-eight sites are all 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).

To validate the quality of source-biased service 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 (similarity) precision as the percentage of the top-10 ranked target services that are considered relevant to the source using $Cosine\_focus_{\sigma}(\tau)$. Relevance judgments were determined by the consensus opinion of three volunteers. Figure 6 shows the precision for the three probers after extracting 40 documents per target service. *Source Bias* results in the highest precision in nine of ten cases, tying with the next best prober in only two cases. 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

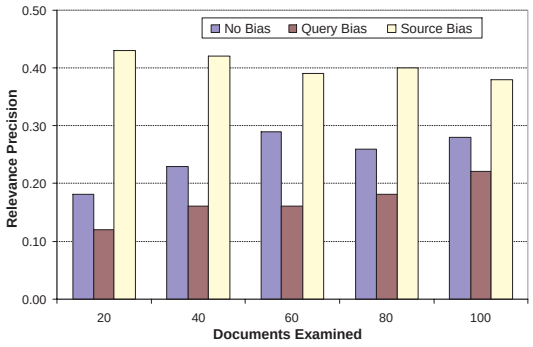

Figure 7: Average Relevance Precision

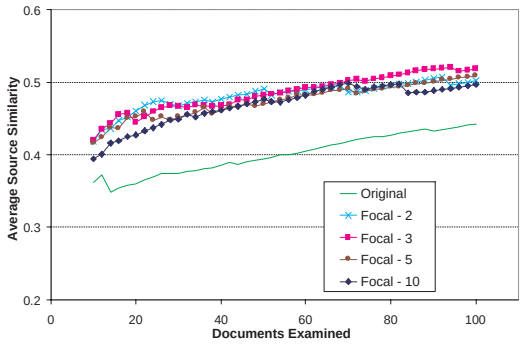

Figure 8: Impact of Focal Term Probing

the query-biased and unbiased probers in all cases considered, especially when based on very few documents.

5.3 Identifying Interesting 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-service relationships that the alternative schemes do not. Unlike the query-biased and unbiased 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.

We first illustrate relationship sets for PubMed over the web collection. In Table 3 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. Again we note, that our interest here is to illustrate the power of the λ-formalation; we leave the optimization of λ-values to future work. In contrast to the simple relevance ranking in Table 2, we see how the source-biased framework can differentiate between the very similar services (the λ-equivalent sites) and the more general services (the λ-superset sites) relative to PubMed. In addition, we can identify sites with some common data (the λ-overlap sites) and sites concerned with significantly different topics (the λ-complement sites).

Similarly, we show in Table 4 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. Again, by relying on BASIL's source-biased analysis we may characterize relationships sets for each source.

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 telecommunications. Interestingly, we can see that there are several overlapping relationships between newsgroups in related but slightly different fields (e.g. `volleyball` and `cricket`). Finally, we also identify several unrelated newsgroups, including `comp.sys.mac.system` relative to `misc.immigration.usa`.

5.4 Probing with Focal Terms

In our final 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 2, 3, 5, and 10 focal term groups from which to draw source-biased probes. 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 service evaluation when considering sources from the single-topic newsgroup collection. [Due to space limitations, we omit these results here].

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.

6. RELATED WORK

Researchers have previously explored different aspects of the service discovery problem, ranging from discovery in a federated environment [25], to identifying services that meet certain quality-of-service guarantees [13], to evaluating services based on a distributed reputation metric [26], to other quality metrics like in [32]. In contrast, we focus on the data relationships between services to power efficient discovery and ranking.

Other researchers have previously studied the problem of repeatedly querying an unknown database in an effort to generate a summary of the database internals [11, 3, 30, 4, 8, 27, 14, 6]. The main purpose of these techniques is to generate a representative content summary of the underlying database. 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.

More recently, Gravano et al. [12] have introduced an extension to the Callan-style probing technique that relies on a learned set of queries for database classification. Their probing method is effective for classifying web sites into a predetermined Yahoo!-style hierarchy, but 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.

Previous research on grouping terms (as in our source-biased probing with focal terms) has focussed on finding terms that are effective for query-expansion [31, 21] or finding similar terms [24]. Our focal term formulation is similar to that used in [21], though their goal is to find close semantic relationships between terms, unlike our coarse-grained groupings.

Table 3: Source-Biased Analysis: Identifying Relationships Relative to PubMed

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

Table 4: 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.sport.volleyball	rec.sport.cricket	0.47	0.46	λ-overlap
rec.games.go	rec.games.chess.misc	0.50	0.53	λ-overlap
rec.crafts.textiles.sewing	comp.lang.perl.misc	0.35	0.32	λ-complement
comp.sys.mac.system	misc.immigration.usa	0.23	0.36	λ-complement

7. CONCLUSIONS

In this paper, we have presented a novel web service discovery and ranking prototype called BASIL that supports a *personalized* view of data-intensive web services through source-biased focus. BASIL supports personalized discovery requests and relevance reasoning through efficient source-biased probing and source-biased relevance metrics. Concretely, we have shown that BASIL allows us to determine in very few interactions whether a target service is relevant to the source service by probing the target with very precise probes. The biased focus measure allows us to evaluate and rank the services discovered and to identify interesting types of source-biased relationships for a collection of services. Additionally, we have introduced source-biased probing with focal terms as a performance optimization to further improve the effectiveness of the basic source-biased algorithm.

8. REFERENCES

[1] Amazon.com. *Amazon.com Web Services*. http://www.amazon.com/gp/aws/landing.html, 2004.

[2] Ariba. http://www.ariba.com, 2003.

[3] J. Callan, M. Connell, and A. Du. Automatic discovery of language models for text databases. In *SIGMOD '99*.

[4] J. P. Callan and M. E. Connell. Query-based sampling of text databases. *Information Systems*, 19(2):97–130, 2001.

[5] J. Caverlee, L. Liu, and D. Rocco. Discovering and ranking web services with BASIL: A personalized approach with biased focus. Technical report, GIT, 2004.

[6] W. W. Cohen and Y. Singer. Learning to query the web. In *AAAI Workshop on Internet-Based Information Systems*. 1996.

[7] Google. *Google Web APIs FAQ*. http://www.google.com/apis/, 2003.

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

[9] IBM. *Web Services for Life Sciences*. http://www.alphaworks.ibm.com/tech/ws4LS/, 2003.

[10] IBM. *IBM UDDI Business Registry*. www.ibm.com/services/uddi/, 2004.

[11] P. G. Ipeirotis, L. Gravano, and M. Sahami. Probe, count, and classify: Categorizing hidden-web databases. In *SIGMOD '01*.

[12] P. G. Ipeirotis, L. Gravano, and M. Sahami. QProber: A system for automatic classification of hidden-web databases. *ACM TOIS*, 21(1):1–41, 2003.

[13] Y. Liu, A. H. Ngu, and L. Zeng. QoS computation and policing in dynamic web service selection. In *WWW '04*.

[14] W. Meng, C. T. Yu, and K.-L. Liu. Detection of heterogeneities in a multiple text database environment. In *CoopIS '99*.

[15] Microsoft. *.NET*. http://www.microsoft.com/net/, 2003.

[16] Microsoft. *Microsoft UDDI Business Registry Node*. http://uddi.microsoft.com/, 2004.

[17] National Center for Biotechnology Information. *NCBI BLAST*. http://www.ncbi.nih.gov/BLAST/, 2004.

[18] M. P. Papazoglou. Service-oriented computing: Concepts, characteristics and directions. In *WISE '03*.

[19] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[20] ProFusion. http://www.profusion.com/, 2004.

[21] Y. Qiu and H.-P. Frei. Concept-based query expansion. In *SIGIR '93*, pages 160–169, Pittsburgh, US.

[22] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. In *Readings in Information Retrieval*. Morgan Kauffman, San Francisco, CA, 1997.

[23] G. Salton, A. Wong, and C. Yang. A vector space model for automatic indexing. *CACM*, 18(11):613–620, 1971.

[24] H. Schutze and J. O. Pedersen. A cooccurrence-based thesaurus and two applications to information retrieval. *Information Processing and Management*, 33(3), 1997.

[25] K. Sivashanmugam, K. Verma, and A. Sheth. Discovery of web services in a federated registry environment. In *ICWS '04*.

[26] R. M. Sreenath and M. P. Singh. Agent-based service selection. *Journal on Web Semantics (JWS)*, 2003.

[27] A. Sugiura and O. Etzioni. Query routing for web search engines: Architecture and experiments. In *WWW '00*.

[28] UDDI. http://www.uddi.org/, 2004.

[29] W3C Working Group. *Web Services Architecture*. http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/, February 2004.

[30] W. Wang, W. Meng, and C. Yu. Concept hierarchy based text database categorization in a metasearch engine environment. In *WISE '00*.

[31] J. Xu and W. B. Croft. Query expansion using local and global document analysis. In *SIGIR '96*, pages 4–11.

[32] L. Zeng, , B. Benatallah, M. Dumas, J. Kalagnanam, and Q. Z. Sheng. Quality driven web services composition. In *WWW '03*.