Understanding Hidden-Web Traffic from The Perspective of A Metasearcher

Zhenyu Liu, I-Hsuan Wu, I-Lin Lee, Junghoo Cho

Computer Science Department, University of California, Los Angeles, CA 90095
{vicliu, iwu, ilinlee, cho}@cs.ucla.edu

Abstract. Previous study on the characteristics of the Hidden Web has largely been about the *content* residing on Hidden-Web sites. In this paper we study how users *access* the Hidden Web by investigating the Web traffic log recorded at the UCLA Computer Science Department. For this purpose, we first develop a method that automatically analyzes a Web traffic log and identifies the HTTP requests that correspond to the Hidden-Web traffic. We then investigate the Hidden-Web traffic to study the current usage and query patterns to the Hidden Web. Our analysis suggests that the Hidden Web is currently being *under-explored* by the users, given the size of the Hidden Web and its user traffic. Additionally, we observe some interesting query patterns that can be utilized to provide better searching services to the Hidden Web.

1 Introduction

The set of pages that are accessible *only* through search interfaces are often called the *Hidden Web* or the *Deep Web* (e.g., the papers in PubMed[1] and Citeseer[2]). This is in contrast to the *Surface Web* which is the set of pages that can be reached by clicking through static links. Due to their enormous estimated size and potentially high-quality content, the Hidden Web has recently attracted significant attention from the research community. For example, Chang et al.[1] have estimated that there exist around 127,000 to 333,000 Hidden-Web sites on the Internet. Another independent study [2] suggests that the amount of information in the Hidden Web is 400 to 550 times larger than that in the Surface Web.

Our work in this paper is motivated by recent research in building a more effective *metasearcher*. A metasearcher is a program that simplifies the interaction between the user and the Hidden Web. It functions like a "hub" to all the Hidden-Web sites and automatically redirects users' queries to the most relevant Hidden-Web site(s). To collect data for building an effective metasearcher, a number of surveys [1, 2] have been conducted from a "content-oriented" perspective by studying the information residing on Hidden-Web sites. Different from these existing efforts, we take a "access-oriented" perspective in this paper and explore how users access the Hidden Web and what kinds of queries they ask. Our basic approach is to collect the Web traffic log from an organization and design an automatic method to analyze the log. We believe that our analysis of Hidden-Web access pattern will provide valuable data in building a more useful metasearcher and in understanding the characteristics of each Hidden Web site. More specifically, our analysis in this paper is centered around the following aspects:

[1] `http://www.ncbi.nlm.nih.gov/entrez/query.fcgi`
[2] `http://citeseer.ist.psu.edu/`

– **Hidden-Web usage.** How heavily is the Hidden Web used in users' daily Web surfing? That is, how much traffic goes to the Hidden Web, compared to the traffic that goes to the Surface Web? Does the Hidden Web receive the amount of usage proportional to the amount of information it contains, or is it currently under-explored? What are the most heavily-used Hidden Web sites? The answers to these questions help us better understand how significant a role the Hidden Web is playing in today's Web surfing activities.
– **Hidden-Web query pattern.** What kinds of regularities can we observe from the way users issue queries to various Hidden-Web sites? In particular, we are interested in the following two kinds of patterns:
  - **Temporal locality of queries**. We expect some queries representing the current "hot" topics are very likely to be asked again in a short period of time. Given the temporal locality, the metasearcher may "cache" or even "pre-fetch" relevant results from Hidden-Web sites, thus reducing the response time and potentially improving the quality of metasearching.
  - **Topical locality of queries**. Usually a Hidden-Web site has a strong topical focus and most queries issued to the site are relevant to the topic. Thus, if two Hidden-Web sites have similar topical focus, they will get many common queries from the users, while two sites of completely different topical focus will not share many queries. We may exploit this correlation between the queries and the sites to identify clusters of Hidden-Web sites on a particular topic and clusters of queries relevant to the topic.

The prerequisite to these studies is the accurate and fully automated analysis of the Hidden-Web traffic, which is a nontrivial task. For example, how do we define Hidden-Web traffic and how do we precisely separate it from the rest of the Web traffic? Further, how can we extract queries from the log without any human intervention, since presumably every Hidden-Web site has its own query format?

The rest of the paper is organized as follows: In Section 2 we review related works. We then answer the above questions and propose our Hidden-Web-traffic-identification as well as query-extraction methods in Section 3. In Section 4 we study the current Hidden-Web usage, and in Section 5 we explore Hidden-Web query patterns. Section 6 concludes the paper.

2 Related work

Recent Hidden-Web research [1–11] has largely concentrated on enhancing the effectiveness of a metasearcher by exploring the content residing on each Hidden-Web site. In contrast, this paper explores the Hidden-Web access and query patterns.

General searching behavior of Web users has been intensively studied over the past few years. While existing works have largely focused on how Web users query commercial search engines to gain access to the Surface Web, few have investigated how Web users access the Hidden Web. A comprehensive review of the existing works can be found in [12] and here we only highlight a few representative ones. In works of Hoelscher [13], Silverstein et al. [14] and Jansen et al. [15, 16] the following issues have been studied: 1) the general query length, 2) use of boolean operators, 3) query behavior within a session, 4) correlation among single query words, 5) how much percentage of search results is delivered to the user, etc. Beyond simple query statistics, Broder [17] and Rose et al. [18] have studied and classified the intentions behind Web users' queries. Further, Beeferman et al. [19] and Davison et al. [20] have proposed to mine the Web query log to discover clusters of similar queries and similar URL's.

HTTP traffic log has also been intensively explored in the research area of *Web caching*. A comprehensive list of literature articles can be found at `web-caching.com` [21]. Existing Web caching research typically focuses on utilizing HTTP traffic incurred by visiting static Web pages, i.e., Surface-Web traffic, whereas we in this paper concentrate on Hidden-Web traffic.

3 Experimental setup and methods

Our method is to study an organization's Web traffic log and identify Hidden-Web access patterns embedded in the log. For this purpose, we have collected the log from the UCLA Computer Science Department from May 21 2003 till August 6 2003, over a period of 11 weeks. Our log involves 601 Web clients in the department and 65,000 different Web sites from outside. To explain how we analyze the log, in Section 3.1 we first introduce the related terminology. In Section 3.2 we formally define "Hidden-Web traffic." Based on the definition, we propose our method to identify Hidden-Web traffic, a two-stage process described in Section 3.3 and Section 3.4.

3.1 Content of the HTTP traffic log

To better illustrate our method, it is necessary to introduce related HTTP terminology. Readers familiar with the HTTP protocol [22] may skip this section.

Each HTTP message has two components, a header and a body. Due to storage constraints, our log only records only the headers of HTTP requests.

1) **Time**: 1083258012.860049
2) **Method**: GET
3) **Requested URL**: citeseer.ist.psu.edu/cs?adddoc=Yes
4) **Referer**: citeseer.ist.psu.edu/cs

(a) A request generated by clicking a static link

1) **Time**: 1083232999.662768
2) **Method**: GET
3) **Requested URL**: citeseer.ist.psu.edu/cs?q=HTTP +log&submit=Search+Documents&cs=1
4) **Referer**: citeseer.ist.psu.edu/cs

(b) A request generated by submitting a query

Fig. 1. Sample headers of GET requests

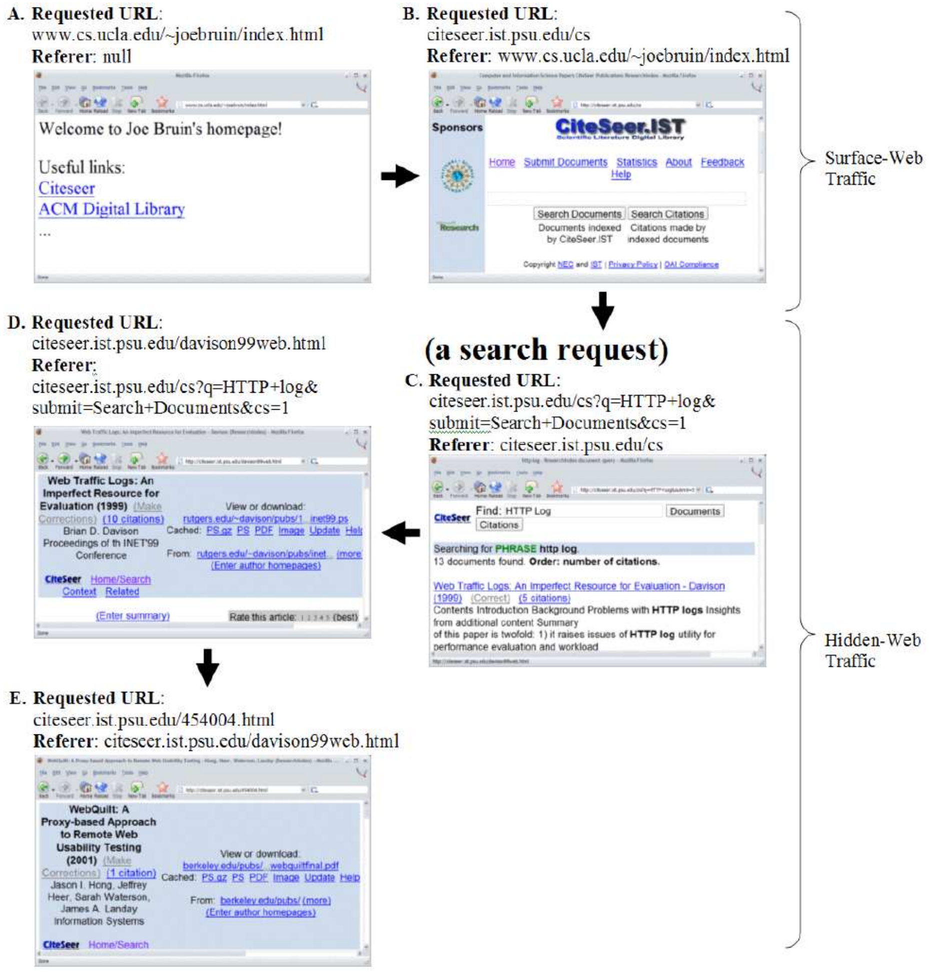
Header content. For each request header we keep 1) the time stamp when the query was issued, 2) the method used by the request, i.e., *GET* or *POST*, 3) the URL on the remote Web site that is being requested, and 4) the URL of the referrer page. Due to privacy concerns, all client-related information (client's IP and port) are anonymized. Figure 1(a) and Figure 1(b) show sample headers that use the GET method.

Referrer. The referrer field will be heavily exploited later by our proposed methods, so we explain it in more detail. The *referrer* of a HTTP request represents a Web page through which the current request is initiated. For example, from the homepage of Citeseer (citeseer.ist.psu.edu/cs), a user clicks on the link "Submit Documents," which results in the request shown in Figure 1(a). Since this request is initiated through Citeseer's homepage, "citeseer.ist.psu.edu/cs" is the referrer.

3.2 Definition of Hidden-Web traffic

We define Hidden-Web traffic as the set of all "search requests" that carry user queries issued to various Hidden-Web sites, plus the requests "spawned" from these "search requests." We illustrate this definition with an example scenario in Figure 2:

In the figure, user Joe Bruin first loads his homepage (request A, the first figure). Once he loads the homepage, he clicks on one of the links "Citeseer," which generates the HTTP request for "citeseer.ist.psu.edu/cs" (request B, the second figure). After loading the Citeseer homepage, he submits a query "HTTP log" through its search box, which generates request C. Among the search results returned by Citeseer, the


A. Requested URL:
www.cs.ucla.edu/~joebruin/index.html
Referer: null

Welcome to Joe Bruin's homepage!

Useful links:
Citeseer
ACM Digital Library
...

B. Requested URL:
citeseer.ist.psu.edu/cs
Referer: www.cs.ucla.edu/~joebruin/index.html

Sponsors
CiteSeer.IST
Scientific Literature Digital Library
Home Submit Documents Statistics About Feedback Help
Search Documents Search Citations
Documents indexed by CiteSeer.IST
Citations made by indexed documents
Research
Copyright NEC and IST | Privacy Policy | OAI Compliance

(a search request)

C. Requested URL:
citeseer.ist.psu.edu/cs?q=HTTP+log&
submit=Search+Documents&cs=1
Referer: citeseer.ist.psu.edu/cs

CiteSeer Find: HTTP Log Documents Citations
Searching for PHRASE http log.
13 documents found. Order: number of citations.
Web Traffic Logs: An Imperfect Resource for Evaluation - Davison (1999) (Correct) (5 citations)
Contents Introduction Background Problems with HTTP logs Insights from additional content Summary
of this paper is twofold: 1) it raises issues of HTTP log utility for performance evaluation and workload

D. Requested URL:
citeseer.ist.psu.edu/davison99web.html
Referer:
citeseer.ist.psu.edu/cs?q=HTTP+log&
submit=Search+Documents&cs=1

Web Traffic Logs: An Imperfect Resource for Evaluation (1999) (Make Corrections) (10 citations)
Brian D. Davison
Proceedings of th INET'99 Conference
View or download: rutgers.edu/~davison/pubs/1...inet99.ps
Cached: PS.gz PS PDF Image Update Help
From: rutgers.edu/~davison/pubs/inet... (more)
(Enter author homepages)
CiteSeer Home/Search Context Related
(Enter summary) Rate this article: 1 2 3 4 5 (best)

E. Requested URL:
citeseer.ist.psu.edu/454004.html
Referer: citeseer.ist.psu.edu/davison99web.html

WebQuilt: A Proxy-based Approach to Remote Web Usability Testing (2001) (Make Corrections) (1 citation)
Jason I. Hong, Jeffrey Heer, Sarah Waterson, James A. Landay
Information Systems
View or download: berkeley.edu/pubs/...webquiltfinal.pdf
Cached: PS.gz PS PDF Image Update Help
From: berkeley.edu/pubs/ (more)
(Enter author homepages)
CiteSeer Home/Search

Fig. 2. A series of requests containing both Surface-Web and Hidden-Web traffic

user clicks on one link to view the literature article he is interested in (request D). The returned article may also contain further links to other articles, and by clicking on the links, Joe reaches another one (request E). We classify request A and B as Surface-Web traffic because they are incurred by following static links. We classify request C as Hidden-Web traffic because it contains the user query. Further, although request D and E are generated through static-link clicking, we classify them as Hidden-Web traffic because they are consequences of searching, "spawned" from request C. Two concerns may arise regarding this definition.

1. Since request B leads to the display of Citeseer's search box, it may be classified as either Surface-Web or Hidden-Web traffic. We feel that viewing a search box is still not a definitive sign that the user has started accessing the Hidden-Web, because that user may further click static links in the page and thus keep herself within the scope of the Surface Web. Therefore in our study we start counting Hidden-Web traffic only when we observe a query-embedded request (such as request C), and classify request B as Surface-Web traffic.

2. The way we classify request D and E might lead to the inclusion of Surface-Web traffic, because a user may navigate from a search-result page (e.g. the result of request C) to some other Surface-Web sites if the page contains a link to the Surface Web. To address this issue, we consider a "spawned" request as part of the Hidden-Web traffic only if the request is sent to the *same* Web site that has spawned it.

Given this definition, we can decompose the task of identifying Hidden-Web traffic into the following two subtasks:

– **Identify the "search request"** (e.g., request C). In Section 3.3 we explain our method to handle this subtask.
– **Identify requests "spawned" from a "search request"** (e.g., request D and E). Section 3.4 presents our method for this subtask.

3.3 Identifying search requests

Formally, we define a *search request* as a HTTP request that carries a user's query, e.g., request C in Figure 2. A search request may use either the GET or the POST method. In this paper, we primarily focus on the requests that use the GET method. Our preliminary investigation indicates that the majority of POST requests are used for authentication purposes (since the site administrators do not want to explicitly show the user ids and passwords as part of a URL, which is the case for the GET method), while many of the search boxes use the GET method.

To identify GET search requests, we need to distinguish them from the rest of the non-search requests, which is a nontrivial task. For example, the requests in Figure 1(b) is a search request, whereas the one in Figure 1(a) is not. However the two requests look very much alike, making it hard for the program to distinguish. We first considered the following two heuristics as a way to distinguish the Hidden-Web traffic, but we found that they led to too many false-positives:

– **Classify based on domain names.** The URL of a search request must at least starts with the domain name of some Hidden-Web site, e.g. citeseer.ist.psu.edu. Thus we might classify requests based on their domain names. However this heuristic will result in too many false-positives because a typical Hidden-Web site also provides static links pointing to some of its Web pages. When users click on these static links, they will incur requests (e.g. Figure 1(a)) that start with a qualified domain name but do not belong to search requests. In addition this heuristic requires compiling a list of Hidden-Web sites, which might not be fully automatic.
– **Classify based on request format.** To carry the user-issued query to the remote Hidden-Web site, a search request follows a special format (e.g. Figure 1(b)): It consists of two parts, divided by a "?" symbol, with the second part being a list of parameters embedding the user's query. However, a great number of non-search requests generated by clicking static links also follow this format (e.g. Figure 1(a)), making this format a non-unique feature to detect search requests. As our results will show, in our dataset only about 4% of the requests following this format are genuine search requests.

To remedy the shortcomings of these heuristics, we design the following automatic two-phase procedure. In phase one, we exploit the format of search requests, same as the second naive heuristic. In phase two, we filter out false-positive cases that pass the test in phrase one. In the following we explain this procedure in more details.

Phase one, detecting candidate search requests with the desirable format. In analyzing the format, we note the URL of a search request consists of two parts:

- **A base URL.** The *base URL* is the part before the "?" symbol, e.g., "citeseer.ist.psu.edu/cs" in Figure 1(b). The base URL corresponds to a search interface on the Hidden-Web site that gets user queries, processes them and dynamically generates search results. Note that it is possible for a Hidden-Web site to host several of such base URL's, each representing a different search function.
- **A parameter list.** This is the part after the "?" symbol, e.g., "q=HTTP+log&submit=Search+Documents&cs=1" in Figure 1(b). A parameter list consists of multiple "⟨parameter name⟩ = ⟨value⟩" pairs, separated by "&" symbols. One of the pairs carries the user-issued query.

```
<search request> ::= <base URL> '?' <parameter list>
<parameter list> ::= <parameter item> {'&' <parameter item>}
<parameter item> ::= <parameter name> '=' <value>
```

Fig. 3. A simple grammar to parse search requests

Given this format, we can design a simple grammar to parse the search requests as shown in Figure 3. In phase one, only those requests whose URL's can be parsed by this grammar will be considered as potential search requests.

Phase two, filtering out false-positives. Requests that pass the test of phase one may be non-search requests generated from static-link clicking (e.g., Figure 1(a)), representing false-positive cases. To filter them out, we propose a "simulated-query-submission" technique as follows: A genuine Hidden-Web search request is generated only when a user interacts with a Web page containing a search box (more precisely, an *HTML form*) and issues a query through that box. Therefore, any search request will have a referrer URL pointing back to the page with the HTML form where the request was originated. Hence, once we find an HTML form in the referrer page, our program can "simulate" submitting a query by pressing the submit button without filling any text boxes. If the HTML form is indeed the from that the user have used, our "simulated request" should have the same format as the user's request. All static URLs embedded in a page that have passed our phase-one test (e.g., Figure 1(a)) would not pass this test, because the request formats would be different.

To implement this criterion, our program first downloads the referrer Web page. We then use HttpUnit [23], an open-source Web page parsing and testing tool, to parse the referrer Web page and find out the HTML forms. For each form we use HttpUnit to "simulate" a submission, without filling in any query terms. This "simulated submission" will generate a "simulated request" which can be captured by our program. If the simulated request matches with the actual request in the log, except that the simulated request has empty parameters (because we have not filled them in during simulation), then we classify the actual request in the log as a search request; otherwise it is a false positive.

Extracting the base URL and the query from a search request. After we identify a search request, we can extract the following two pieces of information critical to our later analysis.

- **The base URL.** Studying the base URL's allows us to observe the subject distribution and popularity of the Hidden-Web sites currently being accessed. The base URL can be easily extracted by using the grammatical rules in Figure 3.

– **The user-issued query.** This is for the study of Hidden-Web query pattern. The query is embedded in the second part, the parameter list (Figure 3). The challenge for query extraction is, from the parameter list alone, a program cannot decide which parameter item contains the query. For example, the three items in Figure 1(b) "q=HTTP+log," "submit=Search+Documents" and "cs=1" looks identical. To solve this problem, we again exploit the extra information provided by the referrer Web page. There is a one-to-one correspondence between 1) each text input box in the referrer Web page to which the user types in a query and 2) each parameter item in the search request carrying the query. Detecting that correspondence allows our program to identify the correct item in the parameter list and further extract the query.

3.4 Identifying "spawned" requests

Given that we are able to detect search requests, our next task is to detect requests "spawned" from a search request (e.g., request D and E in Figure 2). To detect such requests, conceptually we can backtrace the path of referrers, and see whether eventually we hit a search request. In practice, to save computation, we implement a method similar to the standard forward inferencing in executing logic programs (e.g. Datalog) [24].

4 Hidden-Web usage

In this section we first study the scale of current Hidden-Web usage, by quantifying the amount of Hidden-Web traffic. We further investigate the topic and popularity of the Hidden-Web sites that appear in the traffic.

4.1 Number of search requests and base URL's involved

Our traffic log over the 11 weeks contains a total number of 3,556,000 HTTP requests. Using the method described in Section 3.3 we first detect 90249 search requests. In these search requests 511 distinct base URL's are involved. Some base URL's represent general search engines (e.g. Google or AltaVista), not providing the content of any Hidden-Web site, but rather serving as a "gateway" to the Surface Web. Therefore, we further exclude these base URL's and the corresponding search requests by manually compiling a list of general search engines. This exclusion reduces the number of base URL's to 457, and the number of search requests to 8847.

4.2 Hidden-Web usage

Using the method in Section 3.4 we identify 21667 requests "spawned" from the 8847 search requests. Combining these two parts leads to 30514 requests in total, which are counted as the Hidden-Web traffic. Thus, among the 3,556,000 requests in total, 0.86% percent belongs to the Hidden-Web, and the other 99.14% belongs to the Surface Web. This result shows that the Hidden Web is far under-explored considering the amount of information in the Hidden Web — it is estimated that the Hidden Web contains as much information as the Surface Web or even larger [2]. We may have observed this disproportionately small access to the Hidden Web due to one of the following reasons.

– The lack of a convenient, centralized portal to the Hidden Web (analogous to Google in the case of the Surface Web) might have hindered its wide usage. Without convenient central portal like Google, it can be just too cumbersome for Web users to memorize a comprehensive list of Hidden-Web sites that are relevant to the user's information need and visit them individually in their daily Web surfing.

– The poor performance of certain Hidden-Web sites in retrieving relevant information (or the low quality of their contents) might have prevented the popularity of these sites from growing rapidly. Currently the searching capabilities of some Hidden-Web sites are still quite primitive. Given such primitive capabilities, Web users tend to stay within the Surface Web, where they can get their queries answered by general search engines that have implemented much more advanced and sophisticated searching techniques.

We have also studied statistics on the evolution of Hidden-web usage. Roughly, our results show the average amount of Hidden-Web traffic remains constant within the 11 weeks, but there exist significant fluctuations between different times of the day and between different times of the week. For example, we have found the Hidden-Web usage on a weekday is generally twice to three times heavier than that on a weekend day. Within a day, we observe the heaviest Hidden-Web traffic occurs from 12pm to 8pm and the lightest from 4am to 8am.

4.3 Popularity and topics of Hidden-Web sites

In the next task we study which portion of the Hidden Web has been the "hottest." To do that, we quantify the popularity of each Hidden-Web site by the number of queries it receives. We have listed the top-five Hidden-Web sites in terms of the total number of queries and the number of distinct queries they receive in Figure 4 and Figure 5, respectively. As the results show, Citeseer and dictionary Web sites represent the "killing applications" inside the Computer Science Department. A possible reason is users from our specific domain are largely exploring the Hidden-Web to assist their research and paper writing. We also have studied the relationship between each site's popularity and its ranking, and expectedly the relationship exhibits a power law, a characteristic similar to what has been reported about the Surface Web [25].

Base URL	Total # of queries
citeseer.nj.nec.com/cs	954
webster.com/cgi-bin/dictionary	862
dictionary.reference.com/search	671
dictionary.cambridge.org/results.asp	224
yp.yahoo.com/py/ypResults.py	209

Fig. 4. Top base URL's ranked by the total number of queries received

Base URL	# of distinct queries
webster.com/cgi-bin/dictionary	812
citeseer.nj.nec.com/cs	757
dictionary.reference.com/search	637
dictionary.cambridge.org/results.asp	211
thesaurus.reference.com/search	185

Fig. 5. Top base URL's ranked by the number of distinct queries received

4.4 Topic distribution

In the next task, we study the topics of the 457 search interfaces. We use the first two levels of Yahoo! directory as our "taxonomy." We randomly sample one third of the 457 search interfaces and manually identify the topic of each interface. Figure 6 shows the topic distribution of the sampled interfaces. For instance, 11.04% of the search interfaces belong to "Business > Shopping." The top-five topics are highlighted in bold. Note the search interfaces in these five topics may not be the most "popular" ones in terms of receiving the most queries. The latter type of popularity is explored next.

4.5 Popularity of search interfaces

We quantify the popularity of each search interface by the number of queries it receives. To do that, we use the procedure described in Section 3.3 to extract the queries issued

1st level topic	2nd level topic	Distribution
Art	n/a	1.30%
Business	Shopping	**11.04%**
	B2B	**10.39%**
	Finance	5.19%
	Employment	1.30%
Computer	Programming	**5.84%**
	Software	3.90%
	Products	2.60%
Entertainment	Music	5.19%
	Movie	2.60%
Education	Higher Edu	**8.44%**
Government	U.S.	3.25%
News	Newspapers	1.95%
	Web Directory	0.65%
	Weather	1.30%
Health	General	1.30%
Recreation	Travel	3.90%
	Auto	2.60%
Reference	Dictionary	**14.29%**
	Phone & Address	1.30%
Society	Cultural	1.95%
	Religious	1.30%
	Events	0.65%
	Relationships	0.65%

Fig. 6. Topic category distribution of the search interfaces visited

to every search interface. We then rank the search interfaces in a descending order of their popularity, and show the result in Figure 7 and Figure 8. In both figures, the relationships exhibit a power law. In fact, in Figure 7, only 4.6% of the search interfaces receives more than one query per day, whereas 43.1% of the search interfaces receives only one query in the entire period of 11 weeks. We observe the exponent to be 1.38 and 1.33, for the two graphs, respectively.

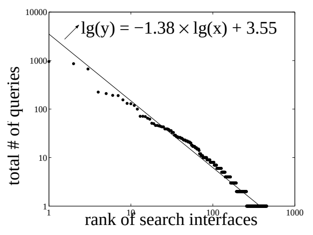


Fig. 7. The total number of queries received by each search interface vs. its rank

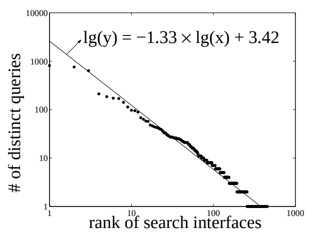
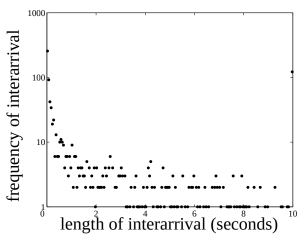


Fig. 8. The number of distinct queries received by each search interface vs. its rank

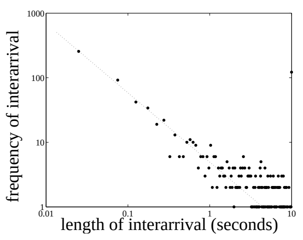
To take a closer look at the data, we listed the top five search interfaces either by the total numbers of queries they receive (Figure 4), or the number of distinct queries (Figure 5). Interestingly, besides Citeseer, the "killer applications" inside the Computer Science Department are Yahoo! yellow page and online dictionaries.

4.6 Query arrival pattern

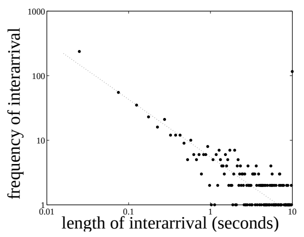
In the next experiment, we verify whether the Poisson process adequately models the query arrival pattern. We know that if the query arrival pattern follows the Poisson process, the query-interarrival time (the length of the interval between two consecutive queries) follows the exponential distribution, and vice versa [26]. To verify, we first identify all the query-interarrival time for the search interface Citeseer. We then show the histogram of the query-interarrival time in Figure 9(a), with the frequency (y-axis) shown in logarithmic scale. We expect to see a linear curve in order to verify the exponential distribution. However, Figure 9(a) fails to exhibit this linear trend. Suspecting the distribution may again follow the Zipf-law, we redraw the data by changing both the x-axis and the y-axis to logarithmic scale, as shown in Figure 9(b). Surprisingly, we observe a linear trend which suggests the power-law distribution fits the data better. To obtain stronger support for our finding, we repeat this process on Webster, and show the distribution in Figure 9(c). Citeseer and Webster are the



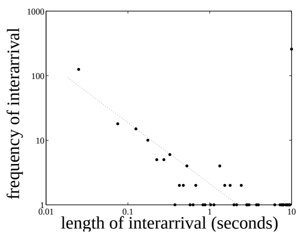
(a) Citeseer, distribution of query-interarrival time, only y-axis shown in log-scale



(b) Citeseer, distribution of query-interarrival time, both x-axis and y-axis shown in log-scale



(c) Webster, distribution of query-interarrival time



(d) 21 search interfaces that receives 20-30 queries, distribution of query-interarrival time

Fig. 9. Distribution of the number of queries received

most "popular" search interfaces and have more sufficient statistics than other search interfaces. To check the distribution on those with less significant statistics, we focus on the group of search interfaces that receive 20 to 30 queries, which is 21 in number. Because of potentially insuffcient data, we merge the distributions generated for each of these 21 search interfaces, as shown in Figure 9(d). In all figures, we observe a strong evidence that queries in our study may not follow the Poisson arrival process. Instead, a power-law distribution seems to better model the query-interarrival time.

5 Hidden-Web query pattern

In this section we study Hidden-Web query patterns, and in particular, the temporal locality and topical locality of queries. We first report general statistics about the query words in Section 5.1, and then explore the two patterns in Section 5.2 and Section 5.3 respectively.

5.1 General statistics

As reported before, from the 11-week log we have extracted 8774 multi-word queries. We segment these multi-word queries into single words using space, comma, '-,' '+' and quotation marks as delimiters. 10298 single words are identified, among which 5853 are unique. For each unique word, we count the times it occurs in the log. We rank the words in a descending order of their frequency, and show the relation of frequency vs. rank in Figure 10. Not surprisingly the distribution exhibits a power law, with the exponent being 0.51. The points in the lower-right flat segment represents words that occur only once (frequency = 1), which are 4093 in number. This accounts for 39.7% among all the 10298 occurrences of words, 69.9% among all the 5853 unique words. The other 1760 unique words reoccur in the log, making up 60.3% of all the occurrences of words.

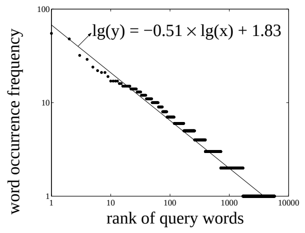


Fig. 10. Word frequency distribution

Word	Frequency
(name, hidden)	55
90024	48
(name, hidden)	32
wireless	29
ca	24
90095	22
computer	21
luke	21
http	19
(name, hidden)	17

Fig. 11. Top-10 words that are frequently-asked

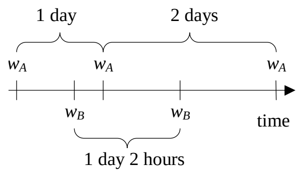


Fig. 12. Sample occurrence intervals for two words w_A and w_B

Figure 11 lists the top 10 words with their frequency. (Stop words [27] such as "a," "the," etc. are filtered out from this list.) Some popular words are the names of individuals in our department, so we choose not to show them. It is not surprising to see UCLA's zip code ("ca 90024" and "ca 90095") appearing in the top-10 list, given Yahoo! Yellow Pages is one of the "killer applications"(Figure 4).

5.2 Temporal locality

We may intuitively expect that the query words issued to Hidden-Web sites may exhibit temporal locality: If a query word is just issued, it is likely to be issued again in the near future. If the temporal locality indeed exists and if a metasearcher can "predict" the queries that are likely to be issued again in the future, the metasearcher may "cache" the results from previous queries or even "pre-fetch" related contents from Hidden-Web sites, thus improving both the quality and response time of metasearching.

To study the temporal locality, we focus on the 1760 reoccurring words that account for 60.3% of all the word occurrences. For each word, we compute the interval between any of its two adjacent occurrences. For example, sample intervals of two words w_A and w_B are shown in Figure 12. We combine the intervals of different words altogether and show the distribution in Figure 13 in linear scale, and in Figure 14 in log scale. The rightmost bars in both figures are significantly larger because they include all intervals larger than 29 days. From the results we can make the following observations:

– The majority of the intervals are short. The 1760 reoccurring words altogether yield 4445 intervals, among which 2316 (52.1%) intervals are shorter than one day (represented by the leftmost bar in Figure 13). Roughly this means the metasearcher expects to see 30% (60%×50%) of the words reoccurring in one day.

– Figure 14 suggests the overall distribution might be resulted from one distribution superimposed upon another: Excluding the leftmost bar, the distribution seems to follow a log-linear trend, suggesting an exponential distribution. On top of this distribution is the distribution of these short intervals (the leftmost bar), representing the "burst arrival" of some "hot" queries.

This result leads to a very interesting future research topic, which is how the metasearcher may accurately predict "hot" queries that frequently reoccur, so that it may perform "caching" and "pre-metasearching" to improve its quality of result.

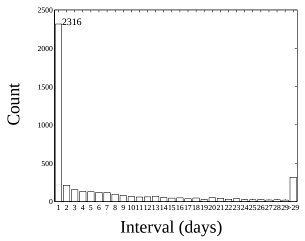


Fig. 13. Interval distribution (linear scale)

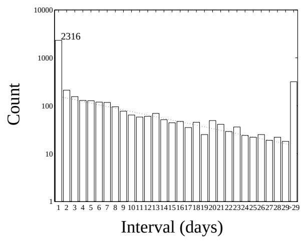
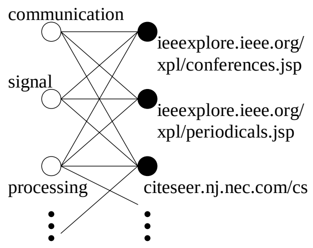


Fig. 14. Interval distribution (log scale)



communication
signal
processing
ieeexplore.ieee.org/ xpl/conferences.jsp
ieeexplore.ieee.org/ xpl/periodicals.jsp
citeseer.nj.nec.com/cs

Fig. 15. A fragment of the bipartite graph

5.3 Topical locality

We often observe multiple Hidden-Web sites that are created with a similar topical focus, e.g., Citeseer and ieeexplore.ieee.org. When users issue queries to these sites, their queries also tend to be related to this topical focus. Detecting this *topical correlation* or *topical locality* embedded in users' Hidden-Web usage allows us to discover similar queries and similar Hidden-Web sites that have a strong correspondence among each other. This helps the metasearcher to better understand the "semantics" of the queries and the corresponding Hidden-Web sites.

This correspondence between queries and Hidden-Web sites can be mathematically explored by formulating a bipartite graph, where the vertices on one side represent all the query words, the vertices on the other side represent all the Hidden-Web sites. An edge is drawn between a query word and a site if the word has been issued to that site. A fragment of this bipartite graph is shown in Figure 15. The fan-out on the left-hand-side is much smaller than that on the right: The number of different sites a word is issued to is much small than the number of different words a site receives. In our data, the maximum fan-out on the left is 16, whereas the maximum on the right is 1154.

To discover the correspondence between words and sites from this bipartite graph, we are particularly interested in the *i-j cores* [28] that are *complete bipartite subgraphs* having i words on the left and j sites on the right. An i-j core indicates a strong correspondence because of the following intuition: When a cluster of, say 3, Hidden-Web sites all receive a common query word, e.g. "processing" in Figure 15, it is still unclear whether these sites share the same topic or they simply receive the same word by chance. However, if additionally these sites all receive two other query words, e.g. "communication" and "signal," we are much more confident that they have the same topical focus. Further, the fact that all these query words are issued to the same set of sites is a reliable evidence that these words represent related topics. The correspondence in this example can be identified by detecting 3-3 cores.

Using a similar algorithm as described by Kumar et al. [28], we have detected all *i*-*j* cores in the bipartite graph, with i ranging from 3 to 10 and j from 3 to 6. The number of these *i*-*j* cores are shown in Figure 16. Note that an *i*-*j* core has many "sub-cores," and such "sub-cores" are not counted for the numbers listed in Figure 16. In other words, the only enlisted 9-3 core ($i = 9, j = 3$) is not a subgraph of the enlisted 10-3 core ($i = 10, j = 3$).

Manually inspecting the detected *i*-*j* cores reveals that the results are highly meaningful. Some *i*-*j* cores with large i or j values are shown in Figure 17. The first core includes computer or communication related query words and sites; The second core is about music and entertainment; The third core includes dictionary-related sites and the words being looked up in these dictionaries; The fourth core includes sites that involves spatial queries. This result suggests that mining the bipartite graph with *i*-*j* core detection is very promising in discovering similar query words, similar sites and the correspondence between the two.

		i (# of query words)							
		3	4	5	6	7	8	9	10
j (# of base URL's)	3	11	3	2	0	4	0	1	1
	4	2	2	0	1	0	0	0	0
	5	1	0	0	0	0	0	0	0
	6	1	0	0	0	0	0	0	0

Fig. 16. Number of *i*-*j* cores

i	j	a set of words (of cardinality i)	a set of base URL's (of cardinality j)
10	3	communication, signal, processing computing, vlsi, systems, ieee computer, aided, design	citeseer.nj.nec.com/cs, ieeexplore.ieee.org/xpl/conferences.jsp, ieeexplore.ieee.org/xpl/periodicals.jsp
9	3	stone, all, well, lopez, frankie ll, madonna, pilots, temple	search.launch.yahoo.com/search/lsearch/video, search.launch.yahoo.com/search/lsearch/all, search.launch.yahoo.com/lsearch
7	3	programmability, emerge, degrade but, prevalent, symmetric, parallelize	webster.com/cgi-bin/dictionary, dictionary.reference.com/search, thesaurus.reference.com/search
3	6	90024, 90025, 90034	autotrader.com/findacar/findacar_form2.jtmpl, movies.yahoo.com/showtimes/showtimes.html, verizonwireless.com/zip/plsql/vzw_zip.zip, weather.yahoo.com/search/weather2, carsdirect.com/used_cars/search, carsdirect.com/build/style

Fig. 17. Sample *i*-*j* cores

6 Conclusion and future work

In this paper we explored the current usage and query pattern of the Hidden Web. Our preliminary study on a 11-week Web traffic log led to the following findings:

- In terms of the amount of usage, the Hidden Web is currently under-explored, and the amount of Hidden-Web traffic is disproportional to the amount of information in it.
- In terms of query pattern, we have observed both temporal locality and topical locality of query words from the log. Such patterns can be exploited by the metasearcher to provide more effective guidance to Web users in accessing the Hidden Web.

In our preliminary study we have excluded POST requests because of the limitation of the dataset. Right now we are recording a more complete HTTP traffic log that permits us to study both GET and POST requests over a longer period of time. We plan to extend our existing study to this enhanced dataset.

References

1. K. Chang, B. He, C. Li, and Z. Zhang. Structured databases on the web: Observations and implications. Technical report, Department of Computer Science, UIUC, 2003.
2. M.K. Bergman. The deep web: Surfacing hidden value. www.brightplanet.com/deepcontent/ tutorials/DeepWeb, 2000.
3. P.G. Ipeirotis and L. Gravano. Distributed search over the hidden web: Hierarchical database sampling and selection. In *Proceedings of VLDB '02*, 2002.
4. P.G. Ipeirotis and L. Gravano. When one sample is not enough: Improving text database selection using shrinkage. In *Proceedings of ACM SIGMOD '04*, 2004.
5. C. Yu, W. Meng, W. Wu, and K.L. Liu. Efficient and effective metasearch for text databases incorporating linkages among documents. In *Proceedings of ACM SIGMOD '01*, 2001.
6. J. Xu and J. Callan. Effective retrieval with distributed collections. In *Proceedings of ACM SIGIR '98*, 1998.
7. L. Si, R. Jin, J.P. Callan, and P. Ogilvie. A language modeling framework for resource selection and results merging. In *Proceedings of CIKM '02*, 2002.
8. J.G. Conrad, X.S. Guo, P. Jackson, and M. Meziou. Database selection using actual physical and acquired logical collection resources in a massive domain-specific operational environment. In *Proceedings of VLDB '02*, 2002.
9. Craswell, P. Bailey, and D. Hawking. Server selection on the world wide web. In *Proceedings of ACM Conf. on Digital Library '00*, 2000.
10. Nobert Fuhr. A decision-theoretic approach to database selection in networked ir. *ACM TOIS*, (3):229 – 259, 1999.
11. Yong S. Choi and Suk I. Yoo. Text database discovery on the web: Neural net based approach. *Journal of Intelligent Information System*, 16:5–20, 2001.
12. B.J. Jansen and U. Pooch. A review of Web searching studies and a framework for future research. *J. of the American Society of Information Science and Technology*, 52(3):235 – 246, 2001.
13. C. Hoelscher. How Internet experts search for information on the Web. In *Proceedings of WebNet '98*, 1998.
14. C. Silverstein, M. Henzinger, H. Marais, and M. Moricz. Analysis of a very large Web search engine query log. *SIGIR Forum*, 33(1):6 – 12, 1999.
15. A. Spink, B.J. Jansen, D. Wolfram, and T. Saracevic. From E-Sex to E-Commerce: Web search changes. *IEEE Computer*, 35(3):107 – 109, 2002.
16. B.J. Jansen, A. Spink, and T Saracevic. Real life, real users, and real needs: A study and analysis of user queries on the Web. *Information Processing and Management*, (2):207 – 227, 2000.
17. A. Broder. A taxonomy of Web search. *SIGIR Forum*, 36(2), 2002.
18. D.E. Rose and D. Levinson. Understanding user goals in Web search. In *Proceedings of the Thirteenth Int'l World Wide Web Conf.*, 2004.
19. D. Beeferman and A. Berger. Agglomerative clustering of a search engine query log. In *Proceedings of ACM SIGKDD '00*, 2000.
20. B.D. Davison, D.G. Deschenes, and D.B. Lewanda. Finding relevant Website queries. In *Proceedings of the Twelfth Int'l World Wide Web Conf.*, 2003.
21. B.D. Davison. Web caching and content delivery resources. http://www.web-caching.com/.

22. World Wide Web Consortium. Hypertext Transfer Protocol - HTTP/1.1. www.w3.org/Protocols/ rfc2616/rfc2616.html, 1999.
23. HttpUnit. httpunit.sourceforge.net.
24. C. Zaniolo. *Advanced Database Systems*, chapter Implementation of Rules and Recursion. Morgan Kaufmann, 1997.
25. A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. Graph structure in the Web. In *Proceedings of the Ninth Int'l World Wide Web Conf.*, 2000.
26. L. Kleinrock. *Queueing Systems*, volume 1. John Wiley and Sons, 1974.
27. G. Salton and M.J. McGill. *Introduction to Mordern Information Retrieval*. McGraw Hill, 1983.
28. R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Trawling the Web for emerging cyber-communities. In *Proceedings of the Eighth Int'l World Wide Web Conf.*, 1999.