Proceedings of the Annual Meeting of the American Society for Information Science and Technology, Providence, RI, November 2004.

The potential of the metasearch engine

Brian D. Davison
Department of Computer Science & Engineering, Lehigh University
19 Memorial Drive West, Bethlehem, PA 18015. Email: davison@cse.lehigh.edu

Research into metasearch engines has traditionally been focused on source engine selection and the re-ranking and integration of multiple search engines' results. In this paper we describe existing and new techniques that metasearch engines can apply to broaden the services provided to the searcher, by capturing and analyzing information already flowing through the meteasearch engine—the queries, their results, and the clicks made by users. In addition, we demonstrate the potential for the engine to generate related queries, related pages, and to recommend queries for web site builders to target through optimization or advertising.

Introduction

Web search engines have been helping users find content online for a decade (McBryan, 1994; Pinkerton, 1994). Today, as then, an individual search engine indexes only a portion of the available content (Bharat & Broder, 1998; Lawrence & Giles, 1998b, 1999a). Metasearch services, introduced a year later (Selberg & Etzioni, 1995; Dreilinger & Howe, 1997), send a user's query to multiple search engines, thus providing the means for a user to search a broader set of documents and potentially get a better set of results (Lawrence & Giles, 1999b).

Building a good metasearch engine can be difficult because different query languages are needed to access various engines and the engines use undisclosed ranking algorithms (Gravano et al., 1997). Popular metasearch engines additionally need to pay for bandwidth, and negotiate with the primary engines for continued high-volume access.

Research into metasearch engines has traditionally been limited to basic functionality (Selberg & Etzioni, 1995, 1997; Dreilinger & Howe, 1997), source engine selection (Dreilinger, 1996; Gauch, Wang & Gomez, 1996; Dreilinger & Howe 1997; Howe & Dreilinger, 1997; Benitez, Beigi & Chang, 1998; Meng et al., 1999; Callan, Connell, & Du, 1999; Craswell, Bailey & Hawking, 2000; Rasolofo, Abbaci & Savoy, 2001), re-ranking and integration (Gauch, Wang & Gomez, 1996; Lawrence & Giles, 1998a; Cohen, Shapire & Singer, 1999; Ng & Kantor, 2000; Dwork et al., 2001; Rasolofo, Abbaci & Savoy, 2001; Montague, 2002; Oztekin, Karypis & Kumar, 2002) and clustering (Zamir & Etzioni, 1998, 1999). This is likely the result of a perceived lack of information available to the metasearch engine. It typically does not know which engine(s) have content relevant to the query, but wants to minimize bandwidth usage and source engine load, and does not have access to the internals of the ranking functions that generated the results, but wants to integrate and thus re-rank the results that it will present to the user. This information poverty may be true of new metasearch engines, but is certainly not the case for successful existing engines, which have a wealth of information that is available to exploit.

In this paper we describe existing and new techniques that metasearch engines can apply to broaden and/or improve the services provided to the searcher. In addition, we demonstrate the potential for the engine to generate related queries, related sites, and recommend queries for sites to target through optimization or advertising.

Information available to a metasearch engine

The real potential of a metasearch engine is defined by the information on which it can act, and its power by the effectiveness of algorithms on that data. As mentioned above, the problems of server selection and results merging have been extensively studied, and in practice, real-world metasearch engines solve it in some fashion. The information used to characterize the source engine for selection can vary, from expert categorization of the source engine (as in ithaki.net), to learned topic coverage based on past result sets (Dreilinger & Howe, 1997), to sampling documents (Callan, Connell & Du, 1999), to explicit self-description using a standard protocol (Gravano, et al., 1997). In contrast, the problem of re-ranking is typically solved primarily using the information provided as part of the result set (e.g., the rankings, and if available, the individual document relevancy scores, from each of the source engines). Less often is the actual document retrieved for this purpose (although see Selberg & Etzioni (1995) and Lawrence & Giles (1998a)) as this can significantly increase the resources required and response time delay.

In most cases, the metasearch engine uses only data local to the query and its results. This paper argues for the collection and long-term maintenance of queries and results over large numbers of queries. In particular, we contend that the collection of user queries, their results, and the

links clicked, constitute a significant base of knowledge that is currently under-exploited by the commercial metasearch engines. This information can be collected and utilized, without going so far as to maintain an index of page contents (the purview of a standalone search engine).

Potential metasearch services

A metasearch engine that captures and analyzes the queries, results, and click-throughs of its user base can create a number of value-added services to distinguish itself from its competitors. These include click-through analysis to improve rankings, evaluating search engine performance, recommending related or expanded queries, suggesting related sites, and competitive web site intelligence. While we mention them here, we will detail the last three and demonstrate their potential in experiments below.

In the late 1990's, DirectHit (acquired by Ask Jeeves in 2000) introduced a technology that tracks which links are selected by searchers. Links that are regularly chosen for a given search term will rise in ranking for that term. Recent research (Joachims, 2002; Oztekin, Karypis & Kumar, 2002) has demonstrated the improvement in results that click-through analysis can provide. In addition, since the typical metasearch engine combines the results of many search engines, click-through analysis can evaluate search engine performance (Selberg & Etzioni, 1995; Joachims, 2002). While not likely a service to many end-users, such a report would likely be of considerable value to search engine companies and financial analysts that follow them.

Search engine users consistently have difficulty producing queries that meet their information needs. One approach to solving this problem is the expansion of queries using terms extracted from documents deemed relevant by a user during a feedback process. That expansion is completed in an attempt to automatically refine users' information needs. Although the concepts supporting automatic query expansion are not new, they continue to be used and have been carried over into the process of query suggestion, in which users are interactively involved in query reformulation. Query suggestion systems, such as those embodied in Teoma's Refine[1] or AltaVista's Prisma (Anick, 2003) functionality, present users with selections of queries that might more readily provide relevant results than the initial user-supplied query. As users select suggested queries they engage in a process of refining their queries until the search engine results meet their information needs. Despite their differences, automatic query expansion and query suggestion have generally relied on textual analysis techniques.

However, more recently introduced query suggestion techniques center on mining the mapping between queries and documents rather than on term extraction from relevant documents (Fitzpatrick & Dent, 1997; Glance, 2000; Beeferman & Berger, 2000; Wen, Nie & Zhang, 2001; Zaiane & Strilets, 2002). These newer techniques suggest that there may be much that can be gleaned from the query to document mappings produced by search engines.

In a typical Web search engine, both textual and link information is recorded and utilized. In recent years, the link information has been exploited extensively to improve the quality of retrieved results. In addition, analysis of the link graph can be used to identify related pages (Gibson, Kleinberg & Raghavan, 1998; Kumar, et al., 1999; Dean & Henzinger, 1999; Flake, Lawrence & Giles, 2000; Reddy & Kitsuregawa, 2001). By exploiting the graph of queries and their results, we can also find and suggest related sites.

Finally, elsewhere (Davison, Deschenes & Lewanda, 2003) we introduce an additional service for competitive web site intelligence that is enabled by the analysis of queries and their results. This service, given a starting URL, would suggest queries for which the URL did not rank highly, but should (because its competitors did). Such a service would be of benefit to web site owners so that they can update their page to be relevant to the query, or possibly decide to buy an advertisement on that query instead.

The query-results graph

Search engines can be thought of as providing a mapping between queries and documents. Additionally, that mapping is often considered to be one-to-many, as, in most cases, many documents are considered relevant by a search engine to a given query. However, that mapping is really many-to-many, as a particular document is often considered relevant to many queries. More specifically, the mapping may be viewed as an directed bipartite graph where a set of queries maps to a set of documents (as shown in Figure 1).

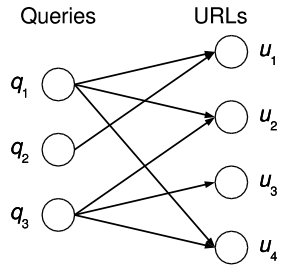


Figure 1: A simple bipartite graph of the query–URL mapping.

[1] http://www.teoma.com/

Mining that graph allows researchers to determine meaningful relationships between queries and between documents as well as between queries and documents. Take, for example, the discovery of related queries and related documents. Both may be achieved through simple operations on the graph, yet both can yield valuable results.

The retrieval of queries related to a user supplied query can be viewed as a traversal of the tree of depth two, with the root being the user supplied query. Using the graph notation presented above, the traversal can be viewed as follows in Figure 2:

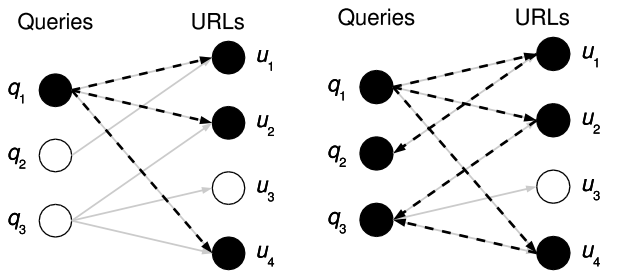


Figure 2: Two step search for related queries.

First we move forward along the edges of the graph starting at the user specified query q_1, reaching each document in the result set of the query (u_1, u_2, and u_4). Starting from those documents we next move backward along the edges of the graph, extending our traversal to those queries that return at least one of the documents reached in the first step (reaching q_1, q_2, and q_3). Thus, queries q_2 and q_3 are considered related to q_1. Retrieving related URLs can be accomplished in an analogous way. See Figure 3 below:

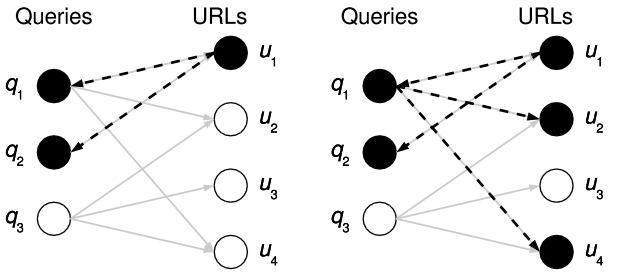


Figure 3: Two step search for related URLs.

Starting at the user specified document u_1, we move backward along the edges of the graph, reaching those queries that return the user specified document as a result (q_1 and q_2). We then move forward along the edges of the graph from each of those queries, reaching those documents in their result sets (finding u_1, u_2 and u_4). Thus, URLs u_2 and u_4 are considered related to u_1.

Although the two algorithms demonstrated above are quite simple they can be combined in ways that produce meaningful tools for competitive Web site intelligence and information retrieval. The four tools that we envision include:

- A mechanism to improve user searching ability. It can provide additional queries that are related to the starting query.
- A mechanism to find related sites, based on other sites that rank highly on the same queries.
- A mechanism for content providers to find queries that do not (presently) rank their site highly, but should.
- A mechanism to find queries that are highly ranked for a given URL.

In the rest of this paper we present some background material and related work. Details on our approach follow, including information about the data set and data structures we use for scalability. We then present examples for each of the applications listed above, discuss some potential concerns, and summarize our findings.

Background

Our approach has strong ties to what might generally be called relationship analysis. In bibliometrics, researchers analyze patterns and relationships of co-citation and bibliographic coupling. Sociologists study social networks among people. In textual data mining, term co-occurrence is often utilized. On the Web, the study of the relationships between pages is typically called link analysis. (Helpful introductions to these topics can be found in (Kleinberg, 1999) and in the popular literature (Barabasi, 2002).) In each of the cases above, one entity is considered related (because of co-occurrence, co-citation, or explicit linkage) to another entity of the same type.

While we apply similar techniques, the direct relationships we examine are between entities of different types (queries and Web pages), which we use to suggest relationships between entities of the same type (as when we find related queries given a starting query, or related pages when given a starting page).

Recent textbooks (Baeza-Yates & Ribeiro-Neto, 1999; Chakrabarti, 2003) provide comprehensive overviews of textual and link analysis approaches to query expansion and suggestion as well as similar document retrieval so the details of those approaches are omitted from this paper. Rather we focus on introducing a number of papers that have pioneered in the application of the query to document mapping to the query suggestion process.

Raghavan & Sever (1995) found that measuring the similarity of query result vectors was better than calculating the query term vector similarity between two

queries. Later, Fitzpatrick & Dent (1997) proposed finding query expansion terms in a pool of documents constructed from the result sets of queries similar to that provided by a search engine user. In their research, query similarity is a measure of the overlap between the result sets of compared queries, weighted by the probability of relevance of each match, giving higher weight to those nearer to the top of the result lists.

Glance (2000) furthers the application of the query to document mapping to the area of query suggestion in her discussion of the Community Search Assistant. In her work, she uses the query to document mapping to construct a graph of queries, where an edge between queries indicates that they are related. Queries with a single document in common are considered related; however, the level of relatedness (degree of overlap) is recorded for ranking purposes. The Community Search Assistant then assists users in the navigation of the query graph by suggesting first-cousin queries as alternatives to a supplied query. Additionally, in constructing graphs from the query results of multiple search engines, Glance makes the observation that the search engine providing the data set has a significant impact on the quality of the structure of the graph and therefore the query suggestions.

Zaiane & Strilets (2002) build upon the concepts introduced in the works above, but consider that equivalent queries might not aid dissatisfied users in refining their information needs. They suggest, in fact, that quasi-similar queries, "queries that yield results that are comparable in content or description," should be preferred. In their work they present seven algorithms that produce quasi-similar queries.

Beeferman & Berger (2000) investigate relationships in the query to document mapping, but only with regard to those mappings that appeared in Lycos[2] click-through records. As in our work, they search for groups of related queries and documents using a content-ignorant approach, but within an agglomerative clustering technique.

Beeferman and Berger subsequently examine how the clusters of queries discovered might be used to enhance the existing query suggestion feature of the Lycos search engine. Similarly, Wen, Nie & Zhang (2001) also employ click-through relationships between queries and documents in the Encarta online encyclopedia to determine clusters of similar queries (to help find that a query is indeed answered in a FAQ).

Jeh & Widom (2002) use a generalized technique to calculate incrementally the similarities (based on a bipartite graph structure) of all pairs of objects (not necessarily of the same type).

Haveliwala et al. (2002) compare multiple document representations for similarity search, and evaluate them using human-built hierarchies such as the Open Directory Project.[3]

Our approach

Our work proceeded in three steps: the collection and construction of a data set, the indexing of that data set and then an analysis of that data set. In this section we describe the first two steps.

Constructing the data set

The building of the query to document mapping started with a trace of queries from the Excite[4] search engine that was collected on December 20, 1999. This trace has been used by many others for query analysis (e.g., Jansen & Spink (2000)). It contains almost 2.5 million requests recorded over an eight hour period. All queries in the trace were made lower-case, but were otherwise unmodified.

Table 1: The top twenty queries in our 1999 Excite data set.

Frequency	Query
5024	sex
3634	yahoo
1661	chat
1649	pokemon
1603	porn
1485	horoscopes
1390	britney spears
1369	mp3
1343	games
1319	weather
1318	hotmail
1290	maps
1283	sitescope test
1196	christmas
1190	www.yahoo.com
1180	yahoo.com
1145	ebay
1118	recipes
1021	horoscope
989	jokes

After having cleaned the trace of queries, we began collecting document data from Google.[5] For each of the queries in the query trace (in decending order by query frequency), we accessed Google via its API[6] recording the URLs of the top ten documents returned. In total, we have recorded results for 430,351 unique queries, generating

[2] http://www.lycos.com/

[3] http://www.dmoz.com/

[4] http://www.excite.com/

[5] http://www.google.com/

[6] http://www.google.com/apis/

3,177,721 unique document URLs. While this is only about one third of all of the unique queries, it accounts for all queries with frequency greater than one, and many queries that were requested only once.

Table 1 lists the top twenty most-frequent queries in our Excite request trace. In Table 2, we show the most frequently occurring URLs within the query results that we obtained from Google.

Table 2: The top twenty URLs (after filtering out 37 sexually-themed sites), ranked by the number of times they appear in Google's top ten results of the (unique, unweighted) queries in our query set.

Frequency	URL
632	http://free.bluemountain.com/
463	http://www.anywho.com/
401	http://www.new-tradition.org/
385	http://people.yahoo.com/
384	http://www.switchboard.com/
356	http://start.earthlink.net/
329	http://123greetings.com/
318	http://www.edmunds.com/
306	http://greetings.yahoo.com/
288	http://www.kbb.com/
278	http://www.whowhere.com/
276	http://www.mp3.com/
258	http://www.egreetings.com/
257	http://www.yahoo.com/
252	http://www.microsoft.com/
230	http://www.mapquest.com/
223	http://codes.ign.com/
216	http://canada411.sympatico.ca/
210	http://www.regards.com/
208	http://www.hallmark.com/

Indexing the data set

In order to efficiently access our data we make use of established search engine data structures. The queries to documents mapping is broken down into two inverted indices, one that maps a query to its resulting document URLs and another that maps the URL of a document to those queries for which it is a result. To keep the size of those indices to a minimum they were constructed using unique identifiers for both the queries and the URLs. Consequently we also constructed a pair of dictionaries that allowed for translation from those unique identifiers to the original query and URL strings.

In summary, the dictionary mapping of strings to IDs is recorded in a file, and loaded into in-memory hash tables for fast retrieval. For lookup, inverted indices are utilized. A primary file contains one entry per ID. That entry contains, among other things, a pointer to a position in another file where the postings list is stored (e.g., the set of query IDs that have the given URL ID as one of their results). By building custom inverted indices that can span multiple files, we enable fast lookups for any relationship, and we will be able to scale to very large collections of queries and query result sets.

In theory, at run-time we are limited by the need to sort our ranked result sets which could potentially be of arbitrary size, but in practice are not large. In reality, our response time is limited by what information we can place in memory for fast access. This is because the algorithms and data structures we use are pre-calculated and ordered for efficient retrieval.

Applications

Earlier we listed four tools that can be built using our techniques. They are helpful in three scenarios, which we detail below.

Query suggestion

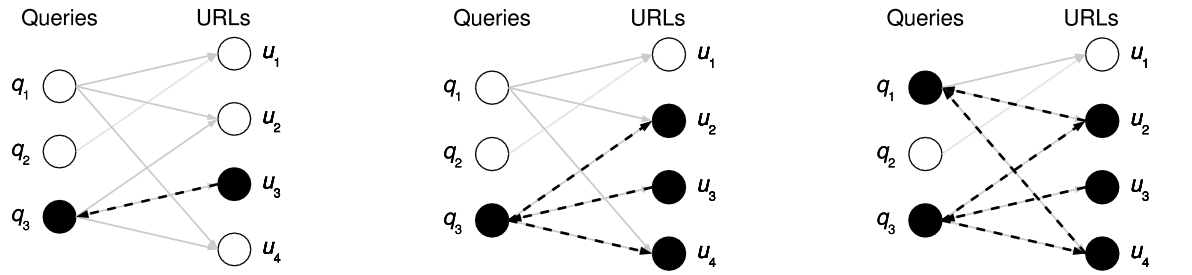
As mentioned earlier, searchers may have difficulty expressing their information need in the form most appropriate to the available data set and content. Therefore, we describe here a general approach that we and others (Fitzpatrick & Dent, 1997; Glance, 2000) use to find related queries.

Our approach is essentially as described in Figure 2. Given a starting query, we find the additional queries that have URLs in common. However, we must also rank those found queries, which we do by the number of URLs that the query has in common with the starting query.

This technique can be embedded within a search engine as an optional, user-selectable action to see the results, or it could be integrated, by presenting the additional URLs that the closely related queries generate.

Table 3: Related queries found when starting with **postal codes**.

"postal codes"
where can i find postal codes
postal code
+postal +service +"zip code"
postal zip codes
zip codes postal
"u.s. postal zip codes"
where can i look up zip codes?
canada post postal code lookup
"u.s. zip codes"



(a) Find associated queries. (b) Find associated URLs. (c) Find queries returning related URLs.

Figure 4: The process of discovering queries that should, but do not, rank a given site highly.

Table 4: Related queries found when starting with **recipies**.

where can i find recipies?
where can i find recipes online
recipies recipes
recipes online
secret recipes
copycat recipes
food allergies
medieval recipes
restaurant recipes
recipe books

Table 5: Related queries found when starting with **medieval recipes**.

renaissance recipes
Renaissance
Saxon
Medieval
a boke of gode cookery
recipies recipes
where can i find recipies?
Recipies
+"renaissance"
renaissance+food

Table 3 presents the related queries generated for **postal codes**, which might remind a potential user that in the United States, the phrase "zip codes" is more common. In Table 4 we see that a misspelled query can sometimes generate useful results. Results for the less common query **medieval recipes** are shown in Table 5, potentially alerting the user to the alternate query "renaissance recipes". Finally, in Table 6 we see more specific queries, such as specifying the interest in free computer games.

Table 6: Related queries found when starting with games.

where can i find games
games?
play computer games
where can i find free computer games?
games to download
www.computer games.com
cool computer games
where can i download computer games
+free+computer+games
where can i find a gaming site.

Table 7: Related sites found when starting with **http://www.mp3.com/**.

http://www.musicmatch.com/
http://www.winamp.com/
http://www.napster.com/
http://www.audiogalaxy.com/
http://www.winamp.com/download/
http://www.kazaa.com/
http://software.mp3.com/software/
http://www.lycos.com/1/?2d
http://www.free-mp3-music-player-downloads.com/
http://www.listen.com/
http://sonique.lycos.com/
http://www.mp3shits.com/
http://www.musiccity.com/
http://www.audiofind.com/
http://launch.yahoo.com/
http://www.artistdirect.com/
http://iomusic.com/
http://mp3.box.sk/
http://www.cdnow.com/
http://www.epitonic.com/

Related sites

A two-step technique similar to the one just described can be used to discover related Web pages. Given a starting URL, we find the queries that rank it highly (in Google's top ten) and the additional URLs in those results. We again rank the list of related sites by the number of queries in common with the starting URL.

Table 8: Related sites found when starting with **http://www.cdnow.com/**.

http://www.cduniverse.com/
http://www.cheap-cds.com/
http://www.mp3.com/
http://www.yesasia.com/
http://www.cdbaby.com/
http://www.musicblvd.com/
http://www.billboard.com/
http://www.music.com/
http://www.sony.com/
http://www.amazon.com/
http://www.music.indiana.edu/music_resources/
http://www.songsearch.com/
http://www.bestbuy.com/
http://www.columbiahouse.com/
http://www.iuma.com/
http://www.clubcd.com/
http://www.allmusic.com/
http://www.gracenote.com/
http://www.buycdnow.com/
http://www.mtv.com/

Table 9: Related sites found when starting with **http://www.weather.com/**.

http://www.weather.com/common/errorpage/errorpage.html
http://www.intellicast.com/
http://www.wunderground.com/
http://www.accuweather.com/
http://weather.yahoo.com/
http://www.cnn.com/WEATHER/
http://www.nws.noaa.gov/
http://espanol.weather.com/
http://cirrus.sprl.umich.edu/wxnet/
http://harvest.weather.com/3com/avantgo/
http://br.weather.com/
http://www.theweathernetwork.com/
http://www.usatoday.com/weather/front.htm
http://www.weatherchannel.com.au/
http://www.bbc.co.uk/weather/

Table 10: Related sites found when starting with **http://www.cnnfn.com/**.

http://www.nasdaq.com/
http://finance.yahoo.com/
http://www.quote.com/
http://www.redherring.com/
http://www.pcquote.com/
http://www.cnn.com/
http://money.cnn.com/
http://quote.yahoo.com/
http://www.bigcharts.com/
http://www.crc.com/
http://europe.cnn.com/
http://www.cnnsi.com/
http://www.nyse.com/
http://asia.cnn.com/
http://go.msn.com/0008/2/nws.asp
http://www.dailystocks.com/
http://www.smartmoney.com/
http://www.investors.com/
http://money.cnn.com/markets/
http://chart.yahoo.com/d

The first four example starting URLs reproduce experiments performed by Haveliwala, et al. (2002). We find high-quality related pages for **http://www.mp3.com/** in Table 7, **http://www.cdnow.com/** in Table 8, **http://ww.weather.com/** in Table 9, and **http://ww.cnnfn.com/** in Table 10, as do Haveliwala et al.

Table 11: Related sites found when starting with **http://www.ucla.edu/**.

http://www.calstatela.edu/
http://www.usc.edu/
http://www.latimes.com/
http://www.ci.la.ca.us/
http://www.berkeley.edu/
http://www.ucdavis.edu/
http://www.nba.com/lakers/
http://www.mta.net/
http://www.lacma.org/
http://www.at-la.com/
http://www.ucsd.edu/
http://www.lazoo.org/
http://www.ucr.edu/
http://www.uci.edu/

As a further example, Table 11 shows sites related to **http://www.ucla.edu/**, which includes a few California universities, the LA Times, and the Los Angeles County Metropolitan Transportation Authority. In contrast, the

related pages for **http://www.edmunds.com/** in Table 12 is made almost entirely of competitor sites.

Table 12: Related sites found when starting with **http://www.edmunds.com/**.

http://www.kar.com/
http://www.nadaguides.com/
http://www.mpta.com/
http://carpoint.msn.com/
http://www.autobytel.com/
http://www.autoweb.com/
http://www.carbuyingtips.com/
http://www.autotrader.com/
http://www.carprices.com/
http://www.cars.com/
http://www.kbb.com/kb/ki.dll/kw.kc.tp?kbb&&32&split
http://www.intellichoice.com/
http://www.edmunds.com/used/
http://www.usedcars.com/
http://go.msn.com/2/1/0/

Competitive web site intelligence

The savvy content provider recognizes the potential for a search engine to send traffic. As a result, content providers will often attempt to optimize particular pages to rank highly in the results of a particular query. By analyzing Web server logs, the content provider is able to determine which queries are successfully sending visitors to the site. However, the content provider has only a narrow view of what queries might be utilized to find the site—the queries found in the web site log and intuition about other possible queries. The content provider does not have a global view of what queries are made, and in particular does not know what relevant queries exist that should rank his site highly, but do not.

Table 13: Suggested queries found when starting with **http://www.hallmark.com/**.

free animated post cards
online greetings cards
birthday "electronic greeting"
christmas e greeting cards
free greetings cards
how can i send a greeting card?
'thank you greetings cards'
electronic cards
birthday e-cards
how can i find web cards to send
new baby greeting card
free virtual greeting cards e-cards

Table 14: Suggested queries found when starting with **http://www.americanexpress.com/**.

where can i apply for a credit card
discover credit card
accept and credit and card and online
visa credit cards
card credit cards
visa credit card
“discover card”
where can I find a credit card?
“credit card”
maps
student credit cards
money

Our system is able to suggest such queries. We start with the same mechanism that finds related Web sites, but extend it one additional step. This process is illustrated in Figure 4. We take the related URLs and find the set of queries that generate them, and remove those queries that also include the starting site. We rank this set of queries primarily by the number of URLs that the query has in common with the set of URLs related to the starting URL.

By comparing the contents of Table 13 with the text on their website, we discover that it might behoove Hallmark to somehow incorporate “greetings”, “electronic”, and “free” into their site, so that it is ranked higher on related queries containing these terms. Similarly, with our system, the corporate site for American Express generates many queries for which the financial services company would be an obvious choice, as well as queries that better match competitors (shown in Table 14).

It may also be helpful in general for a competitive Web site owner to know the important queries for a particular Web page (for which Web server logs are typically not accessible). This service is built into our system, as it is just the single lookup of queries given a site, ranked by query frequency.

Discussion

We readily admit that the sample results described above are anecdotal. However, it is our experience that the quality of the results improves as our data set grows to include the results of more queries. At present our data set is still fairly small—only a third of the more than 1.2 million unique queries have been processed, and we only record 10 results per query. This is an artifact of (reasonable) limitations made by Google for access through their API.

We also note that in time, the cached query results may become out-of-date. At this time we are more concerned with populating our data set than with freshness. A larger-

scale implementation will require tracking and renewing result sets that are likely stale.

In the future, we also hope to add other search engine results to determine the effect that the search engine (with different data sets and ranking algorithms) has on the quality of results from our system. Google tends to rank home pages and popular pages highly, while a system focusing on a textual analysis might generate a different flavor.

Similarly, expanding the search engine results (to 100, for example) from the current 10 would allow for significantly larger related object calculations, and might prompt the use of a URL weighting scheme based on the rank of the URL in the result set.

Summary

Metasearch engines today underutilize information at their disposal. We have identified a number of techniques that can broaden and improve the services provided to the searcher, using information that already flows through the metasearch engine.

By focusing on analyses of relationships between queries and Web pages, we have demonstrated that relatively content-free techniques can provide functions of value. In particular, we have identified the analysis of query to URL relationships as an emerging research area. We also highlighted simple but useful algorithms to find related queries and related URLs, and used them to propose novel applications. With this paper, we have demonstrated the broader applicability of these techniques and built tools of value to web site designers and search engine builders. Given that such techniques are immediately applicable to metasearch engines, we hope that enhanced services can be made available.

ACKNOWLEDGMENTS

This work has been supported in part by NSF grant ANI 9903052. Most of the infrastructure code was written by David G. Deschenes and David B. Lewanda. We also thank others who have provided feedback, ideas, or code, including Matt Brophy, David Kirsch, Yonghui Mou, and Wei Zhang. We additionally thank Dr. Amanda Spink for providing the 1999 Excite query log.

REFERENCES

Anick, P. (2003). Using terminological feedback for web search refinement: a log-based study. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 88–95, Toronto, Canada.

Baeza-Yates, R. & Ribeiro-Neto, B. (1999). *Modern Information Retrieval*. Addison Wesley/ACM Press, New York.

Barabasi, A.-L. (2002). *Linked: The New Science of Networks*. Perseus Publishing.

Beeferman, D. & Berger, A. (2000). Agglomerative clustering of a search engine query log. In *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 407–415.

Benitez, A. B., Beigi, M., & Chang, S.-F. (1998). Using relevance feedback in content-based image metasearch. *IEEE Internet Computing*, 2(4):58–69.

Bharat, K. & Broder, A. (1998). A technique for measuring the relative size and overlap of public Web search engines. In *Proceedings of the 7th International World Wide Web Conference*, Brisbane, Australia.

Callan, J., Connell, M., & Du, A. (1999). Automatic discovery of language models for text databases. In *Proceedings of the ACM SIGMOD Conference*, pages 479–490.

Chakrabarti, S. (2003). *Mining the Web: Discovering Knowledge from Hypertext Data*. Morgan Kaufmann, San Francisco, CA.

Cohen, W. W., Schapire, R. E., & Singer, Y. (1999). Learning to order things. *Journal of Artificial Intelligence Research*, 10:243–270.

Craswell, N., Bailey, P., & Hawking, D. (2000). Server selection on the World Wide Web. In *Proceedings of the 5th ACM Conference on Digital Libraries*, pages 37–46.

Davison, B. D., Deschenes, D. G., & Lewanda, D. B. (2003). Finding relevant website queries. In *Poster Proceedings of the 12th International World Wide Web Conference*, Budapest, Hungary.

Dean, J. & Henzinger, M. R. (1999). Finding related pages in the World Wide Web. In *Proceedings of the 8th International World Wide Web Conference*, pages 389–401, Toronto, Canada.

Dreilinger, D. & Howe, A. E. (1997). Experiences with selected search engines using metasearch. *ACM Transactions on Information Systems*, 15(3):195–222.

Dreilinger, D. E. (1996). Description and evaluation of a meta-search agent. Master's thesis, Colorado State University.

Dwork, C., Kumar, R., Naor, M., & Sivakumar, D. (2001). Rank aggregation methods for the Web. In *Proceedings of the 10th International World Wide Web Conference*, pages 613–622, Hong Kong.

Fitzpatrick, L. & Dent, M. (1997). Automatic feedback using past queries: Social searching? In *Proceedings of the 20th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 306–313, Philadelphia, PA.

Flake, G. W., Lawrence, S., & Giles, C. L. (2000). Efficient identification of web communities. In *Proceedings of the 6th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD-2000)*, pages 150–160, Boston.

Gauch, S., Wang, G., & Gomez, M. (1996). ProFusion: Intelligent fusion from multiple, distributed search engines. *Journal of Universal Computing*, 2(9).

Gibson, D., Kleinberg, J. M., & Raghavan, P. (1998). Inferring Web communities from link topology. In *Proceedings of the 9th ACM Conference on Hypertext and Hypermedia (Hypertext'98)*, pages 225–234.

Glance, N. S. (2000). Community search assistant. In *Artificial Intelligence for Web Search*, pages 29–34. AAAI Press. Presented at the AAAI-2000 workshop on Artificial Intelligence for Web Search, Technical Report WS-00-01.

Gravano, L., Chang, C.-C. K., Garcia-Molina, H., & Paepcke, A. (1997). STARTS: Stanford proposal for Internet meta-searching. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 207–218.

Haveliwala, T. H., Gionis, A., Klein, D., & Indyk, P. (2002). Evaluating strategies for similarity search on the Web. In *Proceedings of the 11th International World Wide Web Conference*, Honolulu.

Howe, A. & Dreilinger, D. (1997). SavvySearch: A metasearch engine that learns which search engines to query. *AI Magazine*, 18(2).

Jansen, B. J., Spink, A., & Saracevic, T. (2000). Real life, real users, and real needs: A study and analysis of user queries on the web. *Information Processing and Management*, 36(2):207–227.

Jeh, G. & Widom, J. (2002). SimRank: A measure of structural-context similarity. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Edmonton, Alberta, Canada. ACM.

Joachims, T. (2002). Optimizing search engines using clickthrough data. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Edmonton, Alberta, Canada. ACM.

Kleinberg, J. M. (1999). Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632.

Kumar, S. R., Raghavan, P., Rajagopalan, S., & Tomkins, A. (1999). Trawling the Web for emerging cyber-communities. In *Proceedings of the 8th International World Wide Web Conference*, Toronto, Canada.

Lawrence, S. & Giles, C. L. (1998a). Inquirus, the NECI meta search engine. In *Proceedings of the 7th International World Wide Web Conference*, Brisbane, Australia.

Lawrence, S. & Giles, C. L. (1998b). Searching the World Wide Web. *Science*, 280(5360):98–100.

Lawrence, S. & Giles, C. L. (1999a). Accessibility of information on the Web. *Nature*, 400:107–109.

Lawrence, S. & Giles, C. L. (1999b). Searching the Web: General and scientific information access. *IEEE Communications*, 37(1):116–122.

McBryan, O. A. (1994). GENVL and WWWW: Tools for taming the Web. In *Proceedings of the 1st International World Wide Web Conference*, Geneva, Switzerland.

Meng, W., Liu, K. L., Yu, C., Wu, W., & Rishe, L. (1999). Estimating the usefulness of search engines. In *Proceedings of Int'l Conference on Data Engineering (ICDE)*, pages 146–153.

Montague, M. (2002). Metasearch: Data Fusion for Document Retrieval. PhD thesis, Dartmouth College.

Ng, K.-B. & Kantor, P. (2000). Predicting the effectiveness of naive data fusion on the basis of system characteristics. *Journal of American Society for Information Science*, 51(13):1177–1189.

Oztekin, B. U., Karypis, G., & Kumar, V. (2002). Expert agreement and content based reranking in a meta search environment using mearf. In *Proceedings of the 11th International World Wide Web Conference*, Honolulu.

Pinkerton, B. (1994). Finding What People Want: Experiences with the WebCrawler. In *Proceedings of the 2nd Int'l World Wide Web Conference: Mosaic and the Web*, Chicago, IL.

Raghavan, V. V. & Sever, H. (1995). On the reuse of past optimal queries. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 344–350, Seattle, WA.

Rasolofo, Y., Abbaci, F., & Savoy, J. (2001). Approaches to collection selection and results merging for distributed information retrieval. In *ACM Conference on Information and Knowledge Engineering (CIKM)*.

Reddy, P. K. & Kitsuregawa, M. (2001). Inferring web communities through relaxed cocitation and dense bipartite graphs. In *Proceedings of the Data Base Engineering Workshop (DEWS 2001)*.

Selberg, E. & Etzioni, O. (1995). Multi-service search and comparison using the MetaCrawler. In *Proceedings of the 4th International World Wide Web Conference*, Boston, MA.

Selberg, E. & Etzioni, O. (1997). The MetaCrawler architecture for resource aggregation on the Web. *IEEE Expert*, 12(1):8–14.

Wen, J.-R., Nie, J.-Y., & Zhang, H.-J. (2001). Clustering user queries of a search engine. In *Proceedings of the 10th International World Wide Web Conference*, Hong Kong.

Zaiane, O. R. & Strilets, A. (2002). Finding similar queries to satisfy searches based on query traces. In *Proceedings of the International Workshop on Efficient Web-Based Information Systems (EWIS)*, pages 207–216, Montpellier, France. Lecture Notes in Computer Science 2426, *Advances in Object-Oriented Information Systems*, Springer-Verlag.

Zamir, O. & Etzioni, O. (1998). Web document clustering: A feasibility demonstration. In *Proceedings of the ACM/SIGIR Conference on Research and Development in Information Retrieval*.

Zamir, O. & Etzioni, O. (1999). Grouper: a dynamic clustering interface to Web search results. In *Proceedings of the 8th International World Wide Web Conference*, Toronto, Canada.