Technical Report

Department of Computer Science
and Engineering
University of Minnesota
4-192 EECS Building
200 Union Street SE
Minneapolis, MN 55455-0159 USA

TR 06-007

Concept-Aware Ranking: Teaching an Old Graph New Moves

Colin Delong, Sandeep Mane, and Jaideep Srivastava

March 20, 2006

Concept-Aware Ranking: Teaching an Old Graph New Moves

Colin DeLong
College of Liberal Arts
University of Minnesota
Minneapolis, MN 55455, USA
delo0041@umn.edu

Sandeep Mane
Department of Computer Science
University of Minnesota
Minneapolis, MN 55455, USA
smane@cs.umn.edu

Jaideep Srivastava
Department of Computer Science
University of Minnesota
Minneapolis, MN 55455, USA
srivasta@cs.umn.edu

ABSTRACT

Ranking algorithms for web graphs, such as PageRank and HITS, are typically utilized for graphs where a node represents a unique URL (Webpage) and an edge is an explicitly-defined link between two such nodes. In addition to the link structure itself, additional information about the relationship between nodes may be available. For example, anchor text in a Web graph is likely to provide information about the underlying concepts connecting URLs. In this paper, we propose an extension to the Web graph model to take into account conceptual information encoded by links in order to construct a new graph which is sensitive to the conceptual links between nodes. By extracting keywords and recurring phrases from the anchor tag data, a set of concepts is defined. A new definition of a node (one which encodes both an URL and concept) is then leveraged to create an entirely new Web graph, with edges representing both explicit and implicit conceptual links between nodes. In doing so, inter-concept relationships can be modeled and utilized when using graph ranking algorithms. This improves result accuracy by not only retrieving links which are more authoritative given a users' context, but also by utilizing a larger pool of web pages that are limited by concept-space, rather than keyword-space. This is illustrated using webpages from the University of Minnesota's College of Liberal Arts websites.

Categories and Subject Descriptors

D.3.3 [Information Storage and Retrieval]: Information Search and Retrieval – *search process, query formulation, information filtering*; H.2.8 [Database Management]: Database Applications – *Data Mining*.

General Terms

Management, measurement, and algorithms.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

KDD'06, August 20–23, 2006, Philadelphia, PA, USA.

Copyright 2006 ACM 1-58113-000-0/00/0004...\$5.00.

Keywords

Concept extraction, Concept-Page Web graph, implicit links, Concept-aware ranking.

1. INTRODUCTION

The increasing use of the World Wide Web and the popularity of web search engines have driven research in improving techniques for web search. A web search can be defined as the ranking of web pages which are most relevant given a user's query. Different approaches, such as PageRank algorithm, HITS algorithm and SALSA algorithm (Brin and Page, 1998; J Kleinberg, 1999; Lempel and Moran, 2000), have been proposed to analyze the link structure of a web graph and then rank the web pages in terms of global authority. Such approaches primarily use only the link structure of the web pages and utilize additional information, such as link text, for keywords (information) related to the web pages. One of the disadvantages of this approach is that web search ranking algorithms on the web graph consider a page having a high rank as an authority (highly relevant page) for all concepts in the webpage, which may not be true. A web page can have multiple concepts associated with it and different levels of authority for each of these concepts. Philosophically, it means that it is difficult for individuals creating the webpages to have perfect knowledge about everything they author. Thus, for example, the homepage of the University of Minnesota's College of Liberal Arts (CLA) has different concepts (multi-word information) like "advising web", and "prospective undergraduate students". Since this homepage is linked to by many other good (authoritative) web pages (other CLA web pages), its ranking is high, but it is not the best source of information on the concept of (say) advising. Additionally, two or more concepts may be related to each other and there is a need to capture such dependencies between concepts when ranking web pages. With effort being made to provide techniques for machine-readable information on the web (Semantic web <http://www.w3.org/2001/sw/>), techniques exploiting this extra information for ranking of web pages are needed.

This paper presents a novel way to incorporate conceptual information in web links for constructing a novel web graph, called a *concept-page graph*. Such a web graph changes the link structure information, which is used by many popular web page

ranking algorithms. Typically, a web graph is viewed as a set of web pages connected to each other via hyper-links. Instead of having a single node for each web page, the proposed approach determines all the concepts associated with each webpage and then splits each web page into multiple nodes. Each such node corresponds to the unique pair of (web page, concept). For identifying the concepts associated with a web page, the anchor (link) text associated with all links from other webpages to that webpage are extracted and concepts are automatically defined. Thus, an approach is proposed for constructing *concept-page graph*. This graph will allow the link structure to capture dependencies between concepts. Such a concept-page web graph can be used with ranking algorithms like PageRank [3], as well as HITS [9]. In addition, we also propose a method to capture implicit links between different webpages having same concept. The motivation is that with billions of web pages on the Web, the possibility of two webpages, with same concept, having a web-link between them decreases. A prototype of with proposed modifications to web graph (i.e.: concept-page graph and implicit links) along with the PageRank algorithm has been implemented. The results of using this approach on the webpages from the University of Minnesota's College of Liberal Arts websites are exceptionally good and further improvements to the prototype are being explored. Thus, the main contributions of this paper are:

- (i) An approach for automated identification and extraction of concepts from the anchor text of web pages is proposed and thus the concepts associated with each web page are identified.
- (ii) A novel approach is proposed to construct the web graph with (w, c) pairs as nodes, where c is a concept associated with a particular web page. This approach captures the dependencies between different concepts in a web graph.
- (iii) A prototype of the proposed approach with PageRank algorithm is available at http://teste.class.umn.edu/search_test.html. Experimental results using this prototype with web page data from the University of Minnesota's College of Liberal Arts website are shown to be very promising.

The rest of the paper is organized as follow -- Section 2 briefly surveys the literature on previous research used for web searches. Section 3 describes the approaches for (i) automated concept extraction from web-graph, (ii) generation of concept-page graph and modifications for implicit links, (iv) concept-aware ranking, and (v) runtime query processing including possible techniques for extraction of concepts relevant to a query (web search) or recommendation. Section 4 provides an overview of the implementation and describes the experimental results. Section 5 concludes this paper with a discussion on future research.

2. BACKGROUND

The increasing use of the World Wide Web and the enormous amount of information available on internet makes web search an important research problem. One of the important tasks of web search is to rank web pages to determine their importance with respect to a user's query. Different ranking approaches have been proposed for assigning such authoritative weights to web pages.

For example, the PageRank algorithm [3] assigns an authority weight to each web page using information about the link structure of the Web. The approach is based on the assumption that a good (authoritative) page is usually pointed to by other good pages and hence must be ranked higher. The HITS algorithm [9] uses a similar approach, but instead uses two vectors of authoritative vectors. This approach works well for queries on broad topics and in case of large number of relevant pages and hyperlinks. Comprehensive surveys of PageRank and related research can be found by Berkhin [1], Bianchini et al. [4] and Langville and Meyer [12], [13].

Each web page is associated with keywords that are found in in-links to that web page. A web page is assumed to be equally knowledgeable of all such keywords related to the web page. Thus, a major limitation of these (and similar) ranking algorithms is that they assume that a web page with high authoritative weight is very knowledgeable of all terms related to it (This is known as *topic drift* [17]). Philosophically speaking, a web page may not be equally informative about all related topics. Initial approaches to address this include heuristic methods for differentially weighting links (Chakrabarti et al [5]; Bharat and Henzinger [2]). Haveliwala [10] proposed a modified PageRank algorithm, called topic-sensitive PageRank. In this approach, ranks are computed separately for each topic and thus each page has a vector of authoritative weights for all topics found in it. The main disadvantage of this approach is that it assumes independence of topics, i.e. a topic is uncorrelated to every other topic and thus separate ranking was assumed to be sufficient. In addition, their approach did not address the issue of topic identification from web graph, instead using topics manually selected from Open Directory Project (<http://www.dmoz.org/>). Richardson and Domingos [17] proposed a query dependent PageRank approach that removed the assumption of the random surfer from the PageRank algorithm. However, for scalability, they compute their query dependent PageRank separately for each term, thus making the independence assumption for terms. As a side note, Richardson and Domingos's was the PageRank analog of Cohn and Hoffman's [7] probabilistic variation for HITS algorithm.

Rafiei and Mendelzon [16] proposed a process to determine the topics on which a web page is considered to be authoritative. Their approach was based on modifying the HITS algorithm for simultaneously ranking web pages as well as identifying topics based on link information. Xue et al [18] showed how implicit relationships between webpages can be captured using user access patterns and then used in ranking algorithms. Other approaches on PageRank computing are related to personalizing search results using user behavior (profile), such as -- Scaled Personalization using user's bookmarks information (Jeh and Widom [11]); user-feedback based authorities list (Chang et al [6]); and Query chains based on sequence of user's search queries (Radlinski and Joachims [15]).

3. APPROACH

Our approach is broken into four steps: *concept extraction*, *building the concept-page graph*, *ranking the concept-page graph*, and *runtime query processing*. First, we must extract

concepts from the raw anchor tag data, which in our case, resides in a MySQL database containing tables of post-process output from CLA's website spider. After the concepts have been generated, the concept-page graph is created using the set of concepts from the previous step. Next, the graph is ranked using PageRank to produce a concept-aware ranking for each webpage/concept combination. Finally, during runtime, queries are processed in order to make use of the ranked concept-page graph and return relevant search results.

3.1 Concept Extraction

At the core of creating a concept-page graph is the initial step: the extraction of concepts from the anchor text data. For this, we use the webpage spider output that powers CLA's search engine. One of the post-process tables used by the spider contains an entry for every recorded link in the following format:

{source_page_id, target_page_id, link_text}

For each unique *{target_page_id, link_text}* pair, we compile an array of the unique source_page_id's associated with that pair. During this process, we ignore links with no anchor text. Not only is there no link text from which concepts can be extracted, but other options such as using the URL as the link text would unfairly tilt the concept extraction in favor of the target, since it is itself mutable by the target, and would make the process less democratic. For the same reason, links with URLs as the anchor text are ignored as well.

For each *{target_page_id, link_text}* pair, we then break the link text into an array of individual words. This is our initial array of concepts, which will eventually contain multi-word concepts, but for now just contain our "bag of words". We then initialize a concept array with word frequencies, which is the number of unique sources for this particular *{target_page_id, link_text}* pair. All possible left-to-right multi-word combinations are then added to the concept array with their frequencies initialized to the current unique source count (as we did with individual words). The resulting array of concepts and frequencies for this target_page_id are then stored in a database. If the concept already exists, then its frequency is incremented by the current unique source count. Additionally, each unique *{source_page_id, target_page_id, concept}* is also stored in the database. This is the core of our concept-page graph.

3.2 Concept-Page Graph Generation

We now have an un-pruned database of target_page_id's, their concepts, and the frequencies for each concept. Since the concepts are "grown" identically for each unique link text string for each target_page_id, it is likely multiple pages will share some of the same concepts. This is, in fact, the intent of having a universal concept growth algorithm. First, however, we must now prune spurious concepts from the database.

```
for each {target_page_id, link_text, frequency, sources} {
  words = get_unique_words(link_text);
  for each {word in words} {
    temp_concepts[word] = frequency;
  }

  store_concepts(temp_concepts, target_page_id, sources);
  temp_concepts = get_multiword_concepts(words, frequency);
  store_concepts(temp_concepts, target_page_id, sources);
}
```

Pseudocode 1: Concept generation

```
function get_multiword_concepts(words, frequency){
  mw_concepts = new Stack(words);
  all_text = words.implode(' ');
  while (mw_concepts.length > 0){
    cand = mw_concepts.pop();
    for each (word in words){
      new_cand = cand + ' ' + word;
      if (
        word.length == 0 ||
        cand.length == 0 ||
        word == cand ||
        cand in word != false ||
        new_cand in all_text == false ||
        processed[new_cand] == true
      ) {
        continue;
      }

      p_mw_concepts[new_cand] = frequency;
      mw_concepts.push(new_cand);
      processed[new_cand] = true;
    }
  }
  return p_mw_concepts;
}
```

Pseudocode 2: get_multiword_concepts function

First, all concepts that occur only once globally (i.e.: for all possible target_page_id's) are removed. The intuition is motivated in part because single-occurring concept references tend to be of extremely low value, but also for performance reasons. Ideally, we want a collection of strong concepts linking different pages together, not a large number of weak concepts that

will exacerbate ranking time computation for almost no gain. A concept that is potentially strong should have at least 2 unique sources utilizing the same concept, which is an initial step towards limiting “concept farming” websites which might attribute concepts to other pages in order to boost their “in context” search result ranking. In several test cases during development, single-occurring concepts often represented over half the total number of concepts for a particular page and almost none were closely related to the target page’s subject matter.

Second, we remove all concepts which are string subsets of longer concepts having the same global frequency. This is because many of the concepts grown in the aforementioned method are not only meaningless, but offer no additional information. When thinking of concepts as connective pieces between pages, we want to maximize the descriptive length of each concept before it starts to lose information. This is, in part, based on association rule generation, where we want to create association rules having the maximum descriptive length as long as its support remains constant.

For example, consider the two concepts in the following table:

Table 1: Concept pruning example

Concept	Frequency
advising	603
advising web	603

Here, “advising” and “advising web” have the same *global* frequency, and because the concepts were grown in the exact same way, we know that the set of source *page_id*’s for both concepts is also the same. Thus, we can remove “advising” because it gives us no additional information. If the frequency of “advising” were higher (and if it were to have a different frequency than “advising web”, it *must* be higher), then we would keep it. This is what is meant by maximizing the descriptive length of a particular concept.

The intuition here is that we want to minimize the storage capacity necessary for the concepts without sacrificing their descriptive strength.

Other heuristics are used for pruning as well. For instance, single-word concepts that are stop words [19], such as “an” or “his” or “awfully”, are removed. Concepts containing these words might not be, however, if the aforementioned descriptive length maximization logic holds. Also, numbers and symbols, often found in link text in pages with a table of contents, are also removed.

Using concepts, instead of a “bag of words” model, is more desirable for several reasons:

1. Multi-word concepts are more discriminating representations of concepts compared to single-word

concepts, as they capture aspects of language which are all but thrown away in a “bag of words” model.

“Academic advising”, for instance, is a more discriminating form of “advising”, but not all advising is purely academic. There is also “career advising”, for instance. If someone were to search for “academic advising”, we would be less likely to pull highly-ranked information on “career advising”, which helps cut down on irrelevant search results for closely-related concepts.

2. Multi-word concepts are often themselves unique concepts due to the incorporation of word order. For example, “student services” is composed of “student” and “services”, but “student services” is itself a unique concept we can model, whereas a “bag of words” model might only consider the co-occurrence of the two words, and not the connective information we have by placing “student” before “services”.
3. Multi-word concepts also allow for the creation of a richer conceptual hierarchy. We can not only infer which single-word concepts are related to other single-word concepts, but also whether or not they have interesting subgroups of multi-word concepts. “Advising” is a good example, as there are several kinds of advising in higher education.

The final step in the construction of our concept-page graph is the addition of implicit links. Up till this point, all of the nodes and edges in the concept-page graph have been generated from explicitly-defined links. Every edge, after all, represents a link from one URL to another URL along a particular concept, itself derived from text within the original anchor tag linking the two URLs. If, however, two URLs share a concept, but are not explicitly-linked, we can add an implicit link to the concept-page graph, described as follows:

Figure 1: Example where $c_i = \text{"advising"}$

Suppose there are two page-concept pairs, $\{A, c_i\}$ and $\{B, c_i\}$. If A and B are not linked to each other explicitly (i.e.: $\{A, B, c_i\}$ or $\{B, A, c_i\}$ do not exist in the concept-page graph), but share the same concept c_i , then we add the “missing” links to the concept-page graph. The purpose of doing this is to fill in gaps where pages are implicitly linked by their shared concepts so that ranking will take into account inter-page conceptual dependencies and, hence, allow for a more accurate ranking of conceptual authorities.

In practice, however, this seems to work better with smaller concept-page graphs, both in terms of improving search results

and the actual computation of a concept-page graph's implicit links. For instance, site-specific spidering of CLA's Student Services website (<http://www.class.umn.edu>) produces a regular web graph of 186 nodes and 2138 edges. Of the 2138 edges, 1120 are to 7 of the top-level web pages for the website. In a site-specific search using only this web graph, since an overwhelming amount of PageRank is attributed to these 7 nodes, these same nodes show up again and again in the search results, even though they may not be particularly authoritative about a given concept, only good hubs. The insertion of implicit links has the effect of bringing conceptual authorities (i.e.: web pages that contain content about a particular concept rather than links to other pages containing content) further up in rankings, since they will have more inlinks than in a purely explicitly-defined web graph. In most contexts, this would seem unnecessary, biasing results towards content-heavy pages rather than just using the global rank (which ranks hubs highly). However, when done in the context of a concept-aware search engine, this is often a desirable trait (especially with smaller graphs). The reason is straightforward: we *want* conceptual authorities in our search results, *not* just hubs that link to conceptual authorities. If the same 7 nodes keep showing up in the search results for a site-specific search, then clearly the value of those results is diminished. For a more in-depth discussion of the value of implicit links in the context of a recommendation engine, see the extended USER technical report [8].

For a large web graph (and therefore, a much larger concept-page graph), the aforementioned problems with conceptual authorities tend to be mitigated. In a large graph, for instance, conceptual authorities tend to be more easily separated from low-value web pages because of deep links from external websites. In a small graph however, there may be only a single link to a high-value conceptual authority, and if low-value nodes in the graph also have a single link to them, then using implicit links helps separate high-value conceptual authorities from low-value web pages.

Moreover, there are serious performance issues when adding implicit links to a large concept-page graph. The size of the entire web graph for CLA's 85 spidered websites is 74,446 nodes and 725,749 edges. The pruned concept-page graph constructed from this web graph contains 314,049 nodes and 1,818,101 edges for some 55,600 distinct concepts. Even when several heuristics were applied to the implicit links calculation query, such as only generating edges where the nodes are from different domains and avoiding concepts which begin with prepositions, over 4,400,000 implicit links were generated, making PageRank over the eventual unique node graph (discussed in the next section) intractable for our experimentation. Addressing scalability issues with respect to implicit links generation/addition is part of our future work.

3.3 Concept-Aware Ranking

Once we have the pruned concept-page graph, the concept-aware ranking process can begin. In our experimental setup, and for this paper, we use PageRank to calculate the authorities of web pages for each concept associated with them (i.e.: for all existing page-

concept pairs). However, several preparation steps are necessary before PageRank can actually be calculated.

Consistent with innumerable papers on PageRank, "sinks", or web pages having no outgoing links, must have outgoing links assigned to them under the "random surfer" model. In concept-aware ranking, however, web pages with no inlinks must have incoming links assigned to them as well. Since each node represents a concept-page pair, pages having no inlinks will have no concepts associated with them and could not be included in the ranking process. As such, we introduce the idea of a null concept.

The process is straightforward. For every page having no inlinks, we randomly generate a source page and create a new link to the page using the null concept. Once all the untargeted pages have been assigned random incoming links, these nodes can now be included in the graph PageRank will run on.

In order for an unaltered version of PageRank or HITS to run using source data from the concept-page graph, we must create a temporary structure for the ranking process using an adjacency matrix. Implemented in a DBMS such as MySQL, we must transform concept-page graph entries of the form:

```
{source_page_id, target_page_id, concept_id}
```

...to the following form:

```
{source_node_id, target_node_id}
```

...where both source_node_id and target_node_id represent unique concept-page pairs.

From the previous step, it is known that every page has at least one concept, even if it is only the null concept. As such, we create a temporary table of concept-page entries and generate a unique node_id for each entry. It is not enough, however, to simply join this table on itself where corresponding source_page_id/target_page_id entries exist in the concept-page graph in order to obtain a sensible adjacency matrix. To do so could inadvertently introduce unnecessary entries into the adjacency matrix by assuming conceptual links between pages that aren't intuitive. As such, the following rule is observed when constructing the adjacency matrix:

If **A** and **B** are web pages having sets of concepts C_A and C_B , and **A** links to **B**, and $C_{B'}$ is the subset of concepts C_B for **A** linking to **B**, then $\{A, C_A\}$ links to $\{B, C_{B'}\}$. See Figure 2.

The intuition here is simple. Page **A** can only confer authority to **B** for the concepts which have been generated from the original anchor tag text which linked **A** to **B** in the first place. To assert otherwise is to say the scope of **A** and **B**'s conceptual relationship is not limited to the concepts for which **A** asserted any authority to, that **A** confers some portion of PageRank to **B** for concepts originating from nodes other than **A**. This would contradict one of our original assertions, that for a particular web page, authority

itself is not a global value, but one that varies from concept to concept. Thus, only the concepts existing in the link from **A** to **B** are used when constructing the adjacency matrix.

Figure 2: Conceptual authority is derived from the referring hub

It is easy to see that the adjacency matrix resulting from the above logic is likely to be large when compared to the original concept-page graph. The aforementioned 1,818,101 edge concept-page graph for CLA, for instance, becomes 8,804,965 edges using this logic. Obviously, this causes PageRank to take longer than it would with a regular web graph (725,749 edges in the case of CLA). It would be much easier, given PageRank’s time complexity, to simply run PageRank on subgraphs pertaining to each individual concept. In doing this, however, we would lose all of the inter-concept relationships (and thus, the authority conferred from a concept to another concept via the links shared by their web pages). Furthermore, for extremely rare concepts, the resulting graphs could have very few edges or fail to have any edges at all (unless implicit links were used, which for a single concept, would create a graph containing nothing but bi-directional links).

After the adjacency matrix has been built, an unaltered version of PageRank can be run to determine conceptual authorities.

3.4 Query Processing

Once the concept-aware PageRank has been calculated for all concept-page pairs, they can be queried through applications such as search engines or recommender systems. For the purposes of this paper, we will only discuss query processing for a search engine and leave recommender system implementation for future work. See the USER technical report for possible integration methods [8].

Figure 3: Concept-aware query process

Search terms (or keywords) entered by the user are first broken into an array of individual words. The keyword array, as well as the original search phrase, is used to query the concept-page graph, which can be thought of as an inverted index of concepts. The results are grouped by page_id, the concept ranks summed for each unique page_id (as pages often match on multiple concepts for a single multi-word query), and returned to the user in the form of search results in descending order of summed concept rank.

4. EXPERIMENTAL RESULTS

In order to better understand how concept-aware ranking performs both in terms of its implementation and search result relevance, a series of experiments were conducted. We group these experiments into three areas: search result quality, graph construction scaling, and ranking time complexity.

All of the data used in our experiments are from the University of Minnesota’s College of Liberal Arts (CLA) websites, 85 of which are spidered on a weekly basis, indexing and retrieving 74,446 unique web pages and 725,749 links (as of this paper’s writing). After sink pages have been addressed, the regular web graph 770,254 edges and 74,446 nodes. For the concept-aware portion, there are two data structures: the concept-page graph and the adjacency matrix used during ranking. The concept-page graph contains 1,818,101 edges, 314,049 nodes, and 55,600 concepts. The adjacency matrix (which in the DBMS, is just a collection of {source, target} pairs) contains 8,804,965 edges and 314,049 nodes.

All experiments were conducted on the same application server and database server using the same DBMS and programming environments. The application server is a Pentium III 1GHz with 4GB RAM and (8) 18.2 GB 10K U160 SCSI hard drives in RAID5 configuration, while the database server is a Dual-Opteron 248 with 4GB DDR400 RAM and (3) 73GB 15K U320 SCSI hard drives in RAID5 configuration. They are connected via a private gigabit network. Our development environment is Linux/Apache/PHP/MySQL (LAMP), running PHP 4.3.10 and MySQL 5.0.18-max.

For both concept-aware ranking and regular ranking (both using PageRank), 10 iterations were used with a damping factor of 0.15. Additionally, both ranking implementations used the same table engines for their temporary data (MEMORY) and persistent data (MyISAM) using identical attribute and index sizes where schema congruencies existed.

The URL of our test search engine, which is selectable between concept-aware search and regular PageRank search, is located at http://teste.class.umn.edu/search_test.html. The **only** metric used to rank the results is each search types' respective PageRank values (either concept-aware or regular). Commonly-used heuristics such as title/phrase weighting are **not** used as we are trying to test **only** the strength of the individual ranking methods.

4.1 Search Result Quality

For measuring search result quality, we queried the spider database (which also tracks queries by users) to find the top 10 most popular queries, which are shown in the table below. These are the search terms we selected for our experiments, as they are the most likely queries to be made by visitors to CLA's websites. The standard precision metric was used for search result performance measurement. Only the first 25 results were considered in the computing the precision values for each query and search type. The results are shown in Table 2 below.

Table 2: Top queries to CLA websites

Top Search Terms	
1.	deans list
2.	majors
3.	scholarships
4.	graduation
5.	orientation
6.	advising
7.	music
8.	study abroad
9.	tuition
10.	psychology

Table 3: Precision values for concept-aware/regular search

Search Terms	Precision (out of 25 results)	
	concept-aware	Regular
advising	96%	16%
dean's list	8%	4%
graduation	100%	8%
majors	100%	24%
music	80%	12%
orientation	20%	8%
psychology	96%	60%
scholarships	96%	12%
study abroad	80%	0%
tuition	20%	40%

As can be seen from the results above, concept-aware search performs much better – drastically so in some cases - than regular search (which, again, is just sorting results on the global PageRank value for the web documents matching the search terms). In general, regular search returned hub pages (i.e.: web pages that are link-heavy) while concept-aware search returned content-heavy pages (though relevant hubs were mixed in as well).

For instance, regular search using the “majors” and “scholarships” terms returned links to the home pages of departments and major-centric academic advising offices, which have navigational links to major and scholarship information. Concept-aware search, on the other hand, was able to return the actual sub-pages referred to by the main page navigation, essentially going a step further than regular search. This pattern was consistent across nearly all the tested search terms.

There was confusion by both concept-aware and regular search for the search term “orientation”, which returned results for Simulink block orientation as well as Freshman/Transfer Student Orientation. Refining the query to “freshman orientation” improves the results for concept-aware search, but the same is not true for regular search, which instead weights results heavily towards URLs mentioning the word “freshman”, but are not necessarily about “freshman”.

The other two cases in which both search methods perform poorly or below average are the search terms “dean’s list” and “tuition”, which is primarily due to the fact that there are only a handful of pages across all of CLA’s websites pertaining to either the Dean’s List or tuition. The Dean’s List, for example, resides in one location (http://www2.cla.umn.edu/news/deans_list.html), while most pages mentioning tuition link to the Office of The Registrar (OTR) for more information. It is interesting to note, however, that concept-aware search only returned five results for this particular query, and every single one was a page *about* tuition, rather than pages linking to OTR for tuition information.

4.2 Graph Construction Scaling

Here we present a comparison of graph size with respect to raw edge/node counts, as well as average out-degree and in/out-degree standard deviations.

Table 4: Graph size

Structure	Edges	Nodes	Concepts
regular web graph	770,254	74,446	NA
concept-page adjac. matrix	8,804,965	314,049	55,600

Table 5: In-degree/Out-degree information

Structure	Avg. Out-degree	StdDev Out-degree	StdDev In-degree
regular web graph	10.3465	24.2418	50.2045
concept-page adjac. matrix	28.0369	70.915	220.215

Clearly, the concept-page adjacency matrix is much larger than the regular web graph in every measure. It has 11.43X the number of edges and 4.2X the number of nodes compared to the regular web graph. Given that we are using concept-page pairs as nodes, rather than just the page by itself, this is not a surprising revelation.

4.3 Ranking Time Complexity

The offline ranking times of each graph type – the regular web graph and the concept-page adjacency matrix – are shown in Table 6.

Table 6: Iteration times (in seconds)

Structure	Avg. Time/Iteration	Total Time
regular web graph	79.92s	799.16s
concept-page adjac. matrix	484.35s	4843.52s

Again, it is unsurprising to see that it takes longer per PageRank iteration for the concept-page adjacency matrix than does for the regular web graph. As the number of nodes grows and in-degree counts grow for each node (when moving from a regular web graph to a concept-page adjacency matrix), iterative computations take longer as well.

It does, however, present a major scalability issue, which is not addressed in this paper. To be implemented on the level of the WWW, the scalability issue would have to be addressed more fully than we have done here, which only include relatively minor

optimizations such as maintaining a temp table for node out-degrees and using main memory tables for intermediate calculations wherever possible.

However, these issues are also faced by present-day search engines, and since we are not altering PageRank itself, advances in speeding up PageRank calculations would be applicable to our system as well. In fact, recent advances in calculating PageRank have shown promise in workload reduction using novel graph partitioning and patch-marking techniques [20]. We hope to include such methods as part of our ongoing work with the system in order to mitigate the scalability issue.

5. CONCLUSIONS

The foundation of this paper was based on the intuition that web pages are not equally authoritative about the concepts they embody, and that the ranking of a web graph must take this into account in order to retrieve web documents relevant to a user’s query. In this paper, we have shown how this approach is possible, that conceptual information can be retrieved from anchor tag data and then leveraged to build and rank a graph that is sensitive to inter-page/concept dependencies.

Initial results from the search engine using these methods are very promising, even surprising. Compared to a regular web graph ranking, a search engine utilizing a ranked concept-aware web graph performed drastically better in our experiments. Search results using the concept-aware technique were often more content-oriented than hub-oriented, bringing the user closer to the relevant content itself and improving overall precision for nearly every search.

There is work is left to be done, however. Scalability is an issue which needs to be more fully addressed, and as such, it is an area we will continue to investigate. Refinements to the concept generation and pruning process are also areas of ongoing exploration, as it would decrease the size of the concept-page graph without sacrificing valuable conceptual information.

6. ACKNOWLEDGMENTS

The authors would like to thank Prasanna Desikan and Nishith Pathak for peppering us with thoughtful questions, helping strengthen our reasoning in countless sections of this paper. Also, Sandeep Mane’s research was supported by NSF Grant No. IIS-0431141.

7. REFERENCES

- [1] Pavel Berkhin. A survey on PageRank Computing. *Internet Mathematics*, Vol. 2 (1): 73–120, 2005.
- [2] K. Bharat and M. R. Henzinger. Improved algorithms for topic distillation in a hyperlinked environment. *Proceedings of the Twenty-First Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1998.

- [3] Sergey Brin and Larry Page. The anatomy of a large-scale hypertextual Web search engine. In *Proceedings of the Seventh international Conference on World Wide Web*, 107-117, 1998.
- [4] M. Bianchini, M. Gori and F. Scarselli. Inside PageRank. *ACM Transactions on Internet Technology*, Vol. 5 (1): 92-128, 2005.
- [5] S. Chakrabarti, B. Dom, D. Gibson, J. Kleinberg, P. Raghavan, and S. Rajagopalan. Automatic resource compilation by analyzing hyperlink structure and associated text. *Proceedings of the Seventh International World Wide Web Conference*, 1998.
- [6] H. Chang, D. Cohn, and A. McCullum. "Learning to Create Customized Authority Lists." In *Proceedings of the 17th International Conference on Machine Learning*, 127-134, 2000.
- [7] D. Cohn and T. Hofmann. The missing link - a probabilistic model of document content and hypertext connectivity. In T. K. Leen, T. G. Dietterich, and V. Tresp (eds.), *Advances in Neural Information Processing Systems*, 13, 2001.
- [8] Colin DeLong, Prasanna Desikan and Jaideep Srivastava. USER (User Sensitive Expert Recommendation): What Non-Experts NEED to Know. *AHPCRC Technical Report TR2005-018*, AHPCRC, Minneapolis, USA, 2005.
- [9] Jon Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, Vol. 46(5): 604-632, 1999.
- [10] Taher Haveliwala. Topic-sensitive pagerank. In *Proceedings of the Eleventh international Conference on World Wide Web*, 517-526, 2002.
- [11] G. Jeh and J. Widom. Scaling Personalized Web Search. Technical Report, Stanford University, 2002.
- [12] Amy Langville and Carl Meyer. Deeper inside PageRank. *Internet Mathematics*, Vol. 1(3): 335-400, 2004.
- [13] Amy Langville and Carl Meyer. A Survey of Eigenvector Methods of Web Information Retrieval. *The SIAM Review*, Vol. 47(1): 135-161, 2005.
- [14] R. Lempel and S. Moran. The Stochastic Approach for Link-Structure Analysis (SALSA) and the TKC Effect. In *Proceedings of the Ninth International Conference on World Wide Web*, 387-401, 2000.
- [15] Filip Radlinski and Thorsten Joachims. Query chains: learning to rank from implicit feedback. In *Proceeding of the Eleventh ACM SIGKDD international Conference on Knowledge Discovery in Data Mining*, 239-248, 2005.
- [16] D. Rafiei and A. Mendelzon. What is this page known for? Computing Web page reputations. In *Proceedings of the 9th international World Wide Web Conference on Computer Networks*, 823-835, 2000.
- [17] Mathew Richardson and Pedro Domingos. The Intelligent Surfer: Probabilistic Combination of Link and Content Information in PageRank. In *Proceedings of the 2001 Neural Information Processing Systems (NIPS) Conference*, 1441-1448, 2002.
- [18] Gui-Rong Xue et al. Implicit Link Analysis for Small Web Search. In *Proceedings of the 26th annual international ACM SIGIR conference on research and development in information retrieval (SIGIR'03)*, 56-63, July 2003.
- [19] MySQL 3.23, 4.0, 4.1 Reference Manual :: 12.7.3 Full-Text Stopwords. <http://dev.mysql.com/doc/refman/4.1/en/fulltext-stopwords.html>
- [20] P. Desikan, Nishith Pathak, J. Srivastava and V. Kumar. "Incremental PageRank Computation on evolving graphs", AHPCRC Technical Report TR- #2004-195, AHPCRC, Minneapolis, USA, 2004.