Finding Related Pages Using the Link Structure of the WWW

Paul - Alexandru Chirita, Daniel Olmedilla and Wolfgang Nejdl
L3S and University of Hannover
Deutscher Pavillon Expo Plaza 1
30539 Hannover, Germany
{chirita, olmedilla, nejdl}@learninglab.de

Abstract

Most of the current algorithms for finding related pages are exclusively based on text corpora of the WWW or incorporate only authority or hub values of pages. In this paper, we present HubFinder, a new fast algorithm for finding related pages exploring the link structure of the Web graph. Its criterion for filtering output pages is "pluggable", depending on the user's interests, and may vary from global page ranks to text content, etc. We also introduce HubRank, a new ranking algorithm which gives a more complete view of page "importance" by biasing the authority measure of PageRank towards hub values of pages. Finally, we present an evaluation of these algorithms in order to prove their qualities experimentally.

1 Introduction

Existing Web search engines only provide restricted services for finding *related pages*. These services are generally based on text corpora of the WWW and return authorities, i.e. pages with high PageRank usually pointed to by many other pages, and especially by many hubs. While authorities could certainly represent important resources, we often also want to learn about the existing hubs for the topic we are interested in. Hubs are Web pages pointing to many sites in one or more domains and represent important overviews of information resources on specific topics. Hubs therefore are often more useful as bookmarks than a set of single specific sites on one specific subject.

The algorithms presented in this paper help users reduce the time spent searching for related pages and introduce a more complete view of page "importance", which considers not only authority, but also hub values. Our main contribution is a new, fast and flexible algorithm (*HubFinder*) for finding hubs (or authorities) related to a given initial set of pages. A related page of any type is one that address the same topic as the original page [5]. Similar to [6], we use the link structure of the WWW as input. If there is a link from page v to page w, then the author of v recommends page w, and the two pages are usually related. However, we will also propose a refinement for our algorithm based on the content analysis solutions proposed in [1]. Furthermore, we have developed a modified version of the PageRank algorithm (*HubRank*), meant to bias the original PageRank values combining both the authority and hub value of each Web page. We use this algorithm to filter the related pages we find. Finally, we have performed several experiments in order to evaluate the quality of our two new algorithms.

2 Previous Work

2.1 Ranking Web Pages

PageRank [9] computes Web page scores based on the graph inferred from the link structure of the Web. It is based on the idea that "a page has high rank if the sum of the ranks of its backlinks is high". Given a page p, its input $I(p)$ and output $O(p)$ sets of links, the PageRank formula is:

$$PR(p) = (1-c) \sum_{q \in I(p)} \frac{PR(q)}{\|O(q)\|} + cE(p) \qquad (1)$$

The dumping factor $c < 1$ (usually 0.15) is necessary to guarantee convergence and to limit the effect of rank sinks. Intuitively, a random surfer will follow an outgoing link from the current page with probability $(1-c)$ and will get bored and select a random page with probability c.

HITS [7] computes two different scores for each page of a Web community: a *hub* score and an *authority* score. The algorithm can be split into the following steps: (1) *Select a starting set of pages*, (2) *extend the starting set*, and (3) *calculate the scores*. The second step is described in section 2.2. *Authority weights* a^p and *hub weights* h^p are assigned to each page applying equation 2 until convergence. After each step the values need normalized such that their squares sum to 1.

$$a^p \leftarrow \sum_{q:(q,p)\in Edges} h^q \quad h^p \leftarrow \sum_{q:(p,q)\in Edges} a^q \qquad (2)$$

Finally, **Randomized HITS** [8] solves HITS' community effect problem (many pages from the same site may have very similar scores) by combining its original formulas with the power iteration of PageRank.

2.2 Finding Related Pages

A good set of related pages has to be small, rich in relevant pages and should contain many strong authorities. [7] extracts the top results of a query sent to a search engine and builds a focused sub-graph of the WWW around them by adding all pages pointed to and at most d pages pointing to each page. We call this operation *Kleinberg extension*. In [7] the initial set is extended only once, but for finding related pages the Kleinberg extension should be iteratively applied several times.

The Companion algorithm [5] finds related pages by building a weighted graph around the starting page and then applying a modified version of HITS. Similarly, [1] outputs the related pages by computing relevance weights for nodes relative to the starting point.

3 HubRank

PageRank has demonstrated to be a successful ranking algorithm although it focuses on authority values exclusively. On the other hand, HITS computes hub ranks according only to hub values. However, if a user searches for "travel agency", she would probably be interested in a high-quality hub page with a list of all the travel agencies, result which can be provided by neither one of these algorithms (PageRank output would not be a hub and HITS hub output would not consider authority value). HubRank addresses this problem by combining both approaches into a single score which biases PageRank [9] towards hubs.

The motivating observation is that a page pointing to a good hub is a candidate to have a high hub rank as well. Many times we encounter pages (perhaps good authorities) with only few out-going links, but towards very important hubs. We consider such pages more important than the hubs themselves, because while a hub can cover lots of topics, such a page will usually contain information about the content addressed by the hubs it is pointing to, about the value of their content (e.g. author opinions), etc.

To achieve these hub scores, we modify the PageRank personalization vector (**E**) to consider the out-degree/in-degree of the pages. More intuitively, the random surfer will always prefer pages with a big out-degree when it gets bored. This way, the global importance of the pages will play an important role in defining general scores, as the random surfer will follow the out-going links with a higher probability than the random ones, and on the other hand, the out-degree of pages will always be considered. In PageRank, the vector **E** is a uniform distribution with $\frac{1}{NP}$ in each entry (where NP is the total number of pages). We set the value of each entry i of **E** to $E_i = |O(i)|\frac{NP}{|O|}$ where $|O|$ is the summation of the out-going links over the whole Web graph and $O(i)$ the set of out-links of page i.

Analogously, authority scores can be computed setting the components of the personalization vector to $E_i = |I(i)|\frac{NP}{|I|}$, where $|I|$ is the summation of all the in-degrees of the pages in the Web. However, when computing authority values, one might use a different matrix than the one used in PageRank (the row-out-going links matrix of the Web graph normalized on columns), depending on how much importance needs to be given to hub values and how much to authority values (e.g. the transposed row-out-going links matrix normalized on rows, as in [8]).

4 HubFinder

Introduction. HubFinder is an algorithm for finding hubs related to an initial base set of Web pages. There are many ways in which *related* can be defined, but here we do this using link information (as in [6]). Two pages are related if one is accessible from the other via the link structure of the Web graph (following either in-going or out-going links). The distance (the number of links followed) between two such pages is usually less than six. According to our experiments, if the distance is bigger, the link information becomes insufficient to say that pages are similar in content with a high enough probability. This approach resembles some focused crawler policies such as considering an out-link v of a relevant page u also likely to be relevant [2].

Base Algorithm. Previous algorithms (such as those from [7, 1]) focus on applying the Kleinberg extension once and then computing hub and authority scores for the resulting pages, in several ways. When searching for related hubs (authorities), one often needs to build a larger set of pages before having found enough representative pages. Such a set can only be built by applying the Kleinberg extension several times, because if one starts from a poor hub/authority, the representative pages related to it might be placed quite a few links away. Note that [7] starts from top-ranked pages, which is not always the case for HubFinder. In this scenario, the size of the resulting set might become very big, even if the maximum number of links followed is 6, as we suggested above. In some cases, we discover up to 500,000 pages around a single page and the computation becomes too slow. HubFinder tackles this problem by trimming the pages discovered after each Kleinberg extension.

It has the following algorithm as a basis:

Let Γ be the Base Starting Set of pages whose related hubs (authorities) we are looking for
$\Gamma \leftarrow$ Apply the Kleinberg Extension on Γ
$\Gamma' \leftarrow \Gamma$
For $i = 1$ to σ do:
  $\Gamma'' \leftarrow$ Apply the Kleinberg Extension on Γ'
  Trim Γ'' to contain only *interesting* pages,
  which are *not* contained in Γ
  $\Gamma \leftarrow \Gamma + \Gamma''$
  $\Gamma' \leftarrow \Gamma''$
End For
Trim Γ to contain as many interesting pages as desired
Return Γ

The first particular aspect of this algorithm is to determine *which are the interesting pages*. This depends on the approach we take. We decided to use global ranking scores, but in principle they could be local as well. A page with high global rank is more important (globally) than a page with high local and small global rank. Also, if the latter is accessible from the user's starting set, then it is considered at least partially important by the user. Another variable that needs to be defined is *how many pages we actually keep after a trimming step*. The specific number should be a percent from the existing pages (e.g. the top 5% of the interesting ones). A possible formula for it is depicted and then discussed below.

$$N_{new} = f(N_{old}) * \frac{N_{old}}{100} = \frac{100 - lg(N_{old}) * 10}{1 + \alpha * (D - 1)} * \frac{N_{old}}{100} \quad (3)$$

The variables have the following meaning:

- N_{old} is the number of pages before the trimming step
- N_{new} is the number of pages after the trimming step
- D represents the current distance from the starting set (how many Kleinberg extensions we applied, or, more specific, the maximum distance of links from any page of the starting set to any page of the current set)
- α is a constant called degeneration factor

The main requirements in designing the formula are to consider the number of Kleinberg extensions applied and to generate smaller values for bigger sets and bigger values for smaller sets. The first criterion is tackled by D in the denominator. The further we go exploring the link structure of the Web, the more important a page has to be in order to be kept. Moreover, α is a constant degeneration factor which allows us to set an importance amount for each distance unit. To accomplish the latter criterion, we introduced the decimal logarithm of the old number of pages in the numerator.

Extensions. When looking for hubs, HubFinder can be further filtered after each step to contain only pages with *acceptable out-degree*. We defined acceptable as $Min(2 + Distance, 10)$. The further a page is from the base page, the bigger out-degree it needs in order to be kept. This approach decreases the number of explored pages by several orders of magnitude, whereas the output set is a subset of the output set we obtain using non-filtered HubFinder.

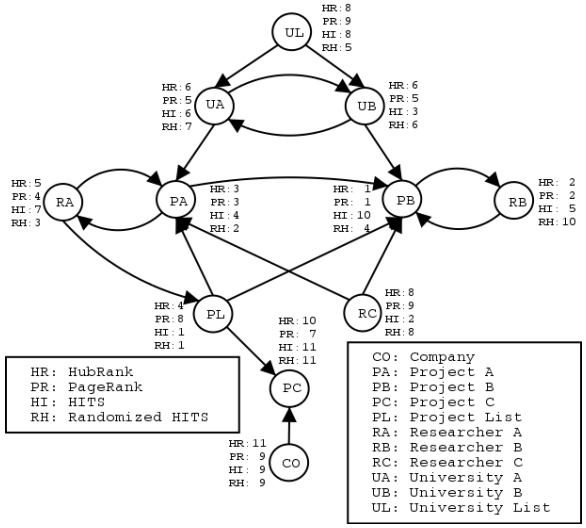

Figure 1. Small graph ranking comparison

Finally, we can improve the quality of HubFinder's results by eliminating the non-relevant nodes (i.e. pages not related to user's query) from the output using content analysis. Let R be the set of results from applying HubFinder to a page p. First, compute the vectors of the first 1000 words from each output document D_j and from p. Then the similarity between p and each D_j can be expressed as in [1]:

$$similarity(p, D_j) = \frac{\sum_{i=1}^{r} w_{ip} * w_{ij}}{\sqrt{\sum_{i=1}^{r} w_{ip}^2 * \sum_{i=1}^{r} w_{ij}^2}} \quad (4)$$

where $w_{ip} = TF_{ip} * IDF(i)$ and $w_{ij} = TF_{ij} * IDF(i)$. TF_{ix} is the frequency of term i in document x and $IDF(i)$ is an estimate of the inverse document frequency of term i on the World Wide Web.

Only those documents from the final set with a similarity value greater than a certain threshold are then returned.

5 Experimental Results

HubRank. Due to space limitations, we will discuss HubRank only on a miniature Web graph, to demonstrate the main effects of the algorithm. Figure 1 summarizes the results of applying HubRank, PageRank, HITS and Randomized HITS on it. We refer the reader to the extended version of this paper [3] for more details.

A good example of the utility of HubRank is the *Project List*. As it is the best hub and a poor authority, HITS and

Randomized HITS rank it in a top position, whereas PageRank does not give much importance to it (although it might be very useful). Finally, HubRank, considering both aspects, ranks it on place 4. Similarly, it also gives a slightly higher rank to *Researcher C* (place 8 out of 11 in HubRank, while it is the last in PageRank).

High authorities with no hub value are likely to have a decreased score. *Project B* is a very good authority and therefore still the first in HubRank, but its score is about 10% lower than the one computed with PageRank and this would result in a small rank decrease with bigger graphs. *Project C* has no hub value at all, which is materialized in HubRank by a rank decrease from 7 to 10 (out of 11) - which we intended.

Generally, we can summarize the outcome of HubRank as follows. A strong authority will always have a good rank, but it will be top ranked only if it is at least an average hub (otherwise it might drop some ranks, compared to PageRank). An average authority will have a rank very dependent on its hub value. Finally, a poor authority will have a low rank if it has a low or average hub value, but it may get a significant raise when it has high hub value (e.g. if one creates a new but very good hub, we try to promote her slightly faster than in PageRank).

HITS and Randomized HITS separate hub values and authority values, which is not useful for us. For example, HITS ranks as second the node *Researcher C* that has no authority at all. On the other hand, PageRank is not considering hub importance at all, which could generate rank drops for important pages, and therefore loss of information. HubRank biases the best authorities of the Web graph according to their value as hubs, so it is more accurate than the other algorithms presented on this example.

HubFinder. We tested HubFinder against two extensions of HITS, in which a starting set of pages is extended several times using the Kleinberg extension and in the end a trimming step is performed.

First, we analyzed our proposed solution for the trimming formula. The number of pages explored by our algorithm is far less than the number of pages explored using the HITS extensions. This also makes HubFinder much faster than all the other algorithms we tested, while all algorithms used similar amounts of main memory. The percentage of explored pages decreases faster at the initial steps and then slower, depending more on the set size as we explore further. In the end, we performed an additional trimming step, in order to adjust the size of the output set to the size we want. Finally, the resulting sets were a bit different in size, but with very similar content.

HubFinder allows personalization by means of different filtering criteria “plugged into” it, depending on the pages one wants to obtain as result. One could for example use HubRank or PageRank to obtain globally appreciated pages. HubRank is also the best criterion when computing input sets for the Personalized PageRank algorithm [4].

6 Conclusions

Most current algorithms for finding related pages are exclusively based on text corpora of the WWW. In this paper, we presented HubFinder, a new algorithm for finding related pages exploring the link structure of the Web graph. Its criterion for filtering output pages is flexible, depending on the user’s interests, and also has different levels: (1) ”pluggable” global page ranks, (2) page out-degree, and/or (3) text content of each output page. Moreover, in our experiments it proved to be faster than the other algorithms we implemented, but provided results of similar quality. We also presented HubRank, a new ranking algorithm which gives a more complete view of page ”importance” by biasing the authority measure of PageRank [9] towards hub values of pages.

References

[1] K. Bharat and M. R. Henzinger. Improved algorithms for topic distillation in a hyperlinked environment. In *Proceedings of SIGIR-98, 21st ACM International Conference on Research and Development in Information Retrieval*, pages 104–111, Melbourne, AU, 1998.

[2] S. Chakrabarti, M. van den Berg, and B. Dom. Focused crawling: a new approach to topic-specific web resource discovery. In *Proceedings of the 8th International World Wide Web Conference*, 1999.

[3] P.-A. Chirita, D. Olmedilla, and W. Nejdl. Finding related pages using the link structure of the www. Technical report, L3S and University of Hannover. http://www.l3s.de/ chirita/pros/pros.html.

[4] P.-A. Chirita, D. Olmedilla, and W. Nejdl. Pros: A personalized ranking platform for web search. In *Proceedings of the 3rd International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems*, Aug 2004.

[5] J. Dean and M. R. Henzinger. Finding related pages in the World Wide Web. *Computer Networks (Amsterdam, Netherlands: 1999)*, 31(11–16):1467–1479, 1999.

[6] G. Jeh and J. Widom. Simrank: A measure of structural-context similarity. In *In Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2002.

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

[8] A. Y. Ng, A. X. Zheng, and M. I. Jordan. Stable algorithms for link analysis. In *Proc. 24th Annual Intl. ACM SIGIR Conference*. ACM, 2001.

[9] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.