Web Search results' ranking: PageRank, HITS and related work

Tejaswi N (04329016),
KReSIT, IIT Bombay

Guide: Prof. Soumen Chakrabarti

November 30, 2004

Abstract

Ranking of results is integral to Web Search due to the large number of pages that potentially satisfy a user query. PageRank and HITS (Hypertext Induced Topic Search) are two popular methods that use eigenvector computations to rank results. In this report we give a detailed overview of these methods and present a few other parameters that affect ranking; these being topic sensitivity, stability, staleness of web content, fine grained sub-page analysis, and link-spamming.

1 Introduction

Web Search is one of the most used features of the Internet today with search engines handling all types of queries from the broad topic based to the really specific ones. Large inverted word indexes are used to serve these queries with popular search engines like Google (http://www.google.com/) indexing more than eight billion pages. Size of a search engine's index is a double edged weapon. With more pages in the index, the chance of having relevant pages to serve specific queries increases. On the flip side, large number of pages returned for a broad topic query will mostly overwhelm normal users. Kleinberg [11] calls these the *scarcity* and the *abundance* problems respectively .Search engines solve the scarcity problem by improving the coverage of their index but need to solve the abundance problem in a different way. Their users typically go through the top twenty results per query before either refining the query or abandoning the search process. This makes it imperative for search engines to rank the results so that the top ranked pages have the "best" quality content. The notion of page quality with respect a query is largely subjective to human judgment. Mechanically extracting the quality of a stand alone page poses a difficult problem. To circumvent this, ranking methods were designed that use the latent hyperlink information of the web to rank pages on their quality. These methods primarily aim to identify the best quality pages using the hyperlink structure of the web.

In 1998, Kleinberg proposed Hypertext Induced Topic Search (HITS) [11] and Google's Brin and Page proposed PageRank [16]. These methods use the underlying hyperlink structure of the web to deduce measures of page quality. They both treat the web as a graph where graph nodes represent pages and graph edges represent hyperlinks connecting pages. HITS attaches an *authority* score and a *hub* score to all the pages in a query-specific subset of the web. The intuition is that authority pages have high quality content relevant to the query (as many hubs link to them) and hub pages have links to many good authorities. Each page has an authority and a hub score, and these are used to construct a ranked list of authorities and hubs that is presented to the user's query.

PageRank gives an offline query-independent global score to each page on the web. This score represents the probability that an aimless surfer visits that page on a random walk of the web. A page's pagerank measures its authority, and is deduced by taking into account the pagerank of all pages that point to it. A page is considered to be authoritative if other authoritative pages link to it. In Section 2 and Section 3 we present a detailed overview of these two ranking methods and in subsequent sections we consider a few parameters on which they can be improved.

In Section 4, we study query-topic sensitive ranking. HITS's query-dependent nature implicitly makes it topic sensitive to each query whereas PageRank, being a global ranking method, needs to incorporate some amount of query dependence to bring in topic sensitivity. In this section we study Haveliwala's Topic Sensitive PageRank [8].

In Section 5, we study the stability of these ranking methods in the face of minor link changes on the web. Ng et al. [14, 15] formally analyze the stability of HITS and PageRank using tools from matrix perturbation theory and Markov chain theory. They broadly conclude that HITS is more vulnerable to minor link changes than PageRank. They also propose two modifications to HITS called Randomized HITS (which incorporates ideas from PageRank) and Subspace HITS, both of which are relatively more stable than HITS.

In Section 6 we study how to incorporate dead links and dead web-neighborhoods into ranking methods. Bar-Yossef et al. [2] study the problem of identifying dead links during crawling and formalize the notion of web-decay. Eiron et al. [6] propose modifications to PageRank that that consider stale pages during ranking.

In Section 7, we study ways of incorporating subpage analysis into ranking. Chakrabarti et al.[5, 3] propose DomTextHits, which uses a page's markup tags, hyperlink structure, and textual content during ranking.

In Section 8 we touch upon link-spamming and methods to combat it. In Section 9 we review a few open research areas in this field.

2 PageRank

According to Google's website, PageRank lies at the heart of its software. PageRank tries to solve the abundance problem discussed in the previous section. Each page is ranked with a pagerank value during an offline ranking process that is independent of any search query. At query-time, relevant pages are retrieved from the index, ordered according to their pagerank and presented to the user. It is expected that the offline process assigns pageranks to all pages such that, when query-specific pages are ordered according to their pageranks, the top results are relevant to the query *and* are of high quality. We use the uppercase term "PageRank" to denote the concept and the lowercase "pagerank" to denote the numerical score which is attached to each page.

A page intuitively has high pagerank if it has many in-links. This intuition fails under two conditions. If page A has only one in-link, but from a highly pageranked page B, we might want to increase A's pagerank too. On the other hand, if a page has many in-links from low pageranked pages, we might *not* want to increase its pagerank. These two points, coupled with our primary intuition gives an informal notion of PageRank. A page's pagerank is an aggregation of the pageranks of its in-link pages. Also, a page with k out-links "passes" only $1/k$-th of its pagerank to each of its out-link pages.

We formalize this intuition with a preliminary definition of PageRank. Let p be the page in question and $R(p)$ be its pagerank. Let B_p be the set of pages that point to p. Let $|p|$ be the number of out-links from p.

$$R(p) = \sum_{q \in B_p} \frac{R(q)}{|q|} \tag{1}$$

This equation is recursive, and may be computed by starting with any set of ranks and iterating the computation till it converges. In matrix terminology, let $\mathbf{P}$ represent the adjacency matrix of the web-graph so that:

$$\mathbf{P}_{i,j} = \begin{cases} \frac{1}{|p_i|}, & \text{if } p_i \text{ links to } p_j; \\ 0, & \text{otherwise;} \end{cases}$$

Let $\mathbf{r}$ be the PageRank vector with all values initialized to $1/N$ where N is the total number of pages on the web. The recursive nature of PageRank can be captured by:

$$\mathbf{r}_{i+1}^T = \mathbf{r}_i^T \mathbf{P} \quad \text{for i = 1, 2, 3, } \ldots \tag{2}$$

We now consider the convergence of this Equation as i arbitrarily increases. PageRank can also be thought of as an aimless surfer's random walk on the web-graph. As we are assigning $\mathbf{P}_{i,j} = 1/|p_i|$, we can treat $\mathbf{P}_{i,j}$ as the probability of the surfer going from p_i to p_j. Rows of $\mathbf{P}$ sum up to either zero or one. Rows that sum up to one correspond to nodes that have one or more out-links. Rows that sum up to zero correspond to pages with no out-links. These are called *dangling nodes* and the aimless surfer can get stuck here. We can get around this by removing all the dangling nodes from our graph and then adding them back after ranking. We can also remedy this by allowing equal probability transitions from a dangling node to all other nodes. This implies that the surfer can now jump from these dangling nodes to any other node with equal probability. Formally, $\forall i$, if $\forall j, \mathbf{P}_{i,j} = 0$, then $\mathbf{P}_{i,j} = 1/N$ where N is the order of the web-graph (total number of pages). This modified matrix $\overline{\mathbf{P}}$ represents the row stochastic transition matrix of the web and the aimless surfer's random walk represents a discrete-time Markov chain. But $\overline{\mathbf{P}}$ still does not ensure convergence.

The Ergodic theorem of Markov chains states that a discrete-time Markov chain, with transition matrix $\overline{\mathbf{P}}$, will have exactly one probability vector $\mathbf{r}$ which satisfies $\mathbf{r}^T = \mathbf{r}^T\overline{\mathbf{P}}$ (meaning convergence) if $\overline{\mathbf{P}}$ is irreducible (i.e., there is a directed path from every node to every other node) and aperiodic (means that there exists at least one node for which the transition from that node to itself is possible).

This vector is also called the stationary distribution; it is the eigenvector of the transition matrix, associated with the eigenvalue 1.

The web-graph is neither strongly connected nor aperiodic. So, to ensure convergence, PageRank adds low probability transitions from every node to every other node in the web-graph. The aimless surfer either follows one of the out-links of the current page, or with some low probability takes a random jump out of the current page to some new page. The complete definition of the random walk now has an aimless surfer on the current page with two options - With probability α, randomly choose a page from the web and jump there or with probability $1-\alpha$ randomly choose an out-link from the current page and follow it. After a large number of such transitions each node on the web-graph has an associated value which represents the probability that the surfer is on that node. This value is the working PageRank of the page. This completes the intuitive notion of PageRank.

We now formalize this complete intuition of PageRank. Let $\overline{\mathbf{P}}$ be the row stochastic transition matrix of the web. We choose a constant α between 0 and 1. Let $\mathbf{1}$ represent the unit column vector of required dimensions and let N represent the number of nodes in the graph. We obtain the modified matrix $\mathbf{G}$:

$$\mathbf{G} = \alpha\frac{\mathbf{1}\times\mathbf{1}^T}{N} + (1-\alpha)\overline{\mathbf{P}} \tag{3}$$

Matrix $\mathbf{G}$ is irreducible and aperiodic. This implies that there exists a unique vector $\mathbf{r}$ such that:

$$\mathbf{r}^T = \mathbf{r}^T\mathbf{G} \tag{4}$$

With $\mathbf{G}$ now being a row stochastic matrix, after multiple iterations, $\mathbf{r}$ will converge to its dominant eigenvector[1]. This can be computed using power iterations. Initially $\mathbf{r}$ has values $1/N$ (It can actually have arbitrary initial values). This vector is then used iteratively in the $\mathbf{r}_{i+1}^T = \mathbf{r}_i^T\mathbf{G}$ equation till $\mathbf{r}$ stabilizes. After each iteration, the vector is normalized so that the sum of its elements is 1. After convergence, the i-th component of $\mathbf{r}$ is the pagerank of page i. Value of α is kept between 0.1 and 0.2.

PageRank gives an importance score to each page and this score implies no relevance with respect to queries. In its original form, Pagerank is query independent. Some amount of query dependence can be incorporated into PageRank by replacing $\frac{\mathbf{1}\times\mathbf{1}^T}{N}$ in $\mathbf{G}$ with $\mathbf{1}\times\mathbf{v}^T$ where $\mathbf{v}$ is has designated probabilities for specific pages. This implies that the aimless surfer now jumps (with probability α) to a random page from a specified set, instead of a random page from the whole web. This allows PageRank to either increase or decrease any page's pagerank without affecting anything else. We will study this in more detail in subsequent sections.

[1] also called the principal eigenvector; with eigenvalue 1

3 HITS

Hypertext Induced Topic Search (HITS), due to Kleinberg [11], is a query-specific way of processing a subset of the web to deduce a set of *authorities* and *hubs*. Authority pages have high quality information pertaining to the query and hubs have many links to such authorities. The user query is sent to a system that uses an inverted word index and k pages (called the *root* set) are identified. If N is the total number of pages on the web, then k is chosen such that $k \ll N$. Pages linking from and to the root set pages are also identified to form the *expanded* set. Some fixed number of in-links and out links from each of the root set pages are included in the expanded set. This is to restrict the overall number of pages taking part in the ranking. Union of the root and the expanded sets is called the *base* set of the query. The intuition is that the base set will have good hubs and authorities pertaining to the query. Also, good hubs have out-links to many good authorities good authorities have in links from many good hubs. These intuitions are translated into an algorithm that constructs a web-graph of the base set and gives hub and authority scores to each page. This process is called *topic distillation*.

Let E be the set of hyperlinks in the base set of a query. Let $e_{ij} \in E$ represent the hyperlink between page i and page j. Authority and hub scores of the base set pages are initialized with unit values. They are then refined using hypertextual information of the base set. Let h_i and a_i represent the hub and authority scores of page i.

$$h_i^{t+1} = \sum_{j:e_{ij}\in E} a_j^t \tag{5}$$

$$a_j^{t+1} = \sum_{i:e_{ij}\in E} h_i^t \tag{6}$$

$$\text{for } t = 1, 2, 3, \ldots$$

We notice that there are *two* "reinforcement" equations here. Recursively, authorities are reinforced by hubs and hubs are reinforced by authorities. Stable hub and authority scores for each page are computed by iterating this computation till their values stabilize. In matrix terminology, let

$\mathbf{P}$ represent the adjacency matrix of the base set web-graph so that:

$$\mathbf{P}_{i,j} = \begin{cases} 1, & \text{if } p_i \text{ links to } p_j; \\ 0, & \text{otherwise;} \end{cases}$$

Let $\mathbf{h}$ and $\mathbf{a}$ represent hub and authority score vectors respectively. Now, Equations 5 and 6 can be written as:

$$\mathbf{h}^{t+1} = \mathbf{P} \times \mathbf{a}^t \tag{7}$$
$$\mathbf{a}^{t+1} = \mathbf{P}^T \times \mathbf{a}^t \tag{8}$$
$$\text{for } t = 1, 2, 3, \ldots$$

Substituting Equations 7 and 8 in each other (taking care of the step variable t) we get:

$$\mathbf{h}^{t+1} = \mathbf{P}\mathbf{P}^T \;.\; \mathbf{h}^t \tag{9}$$
$$\mathbf{a}^{t+1} = \mathbf{P}^T\mathbf{P} \;.\; \mathbf{h}^t \tag{10}$$
$$\text{for } t = 1, 2, 3, \ldots$$

Power method (repeated iterations) can be used to solve this system of equations. After each iteration, the vectors are normalized so that their elements sum to 1. Hub vector $\mathbf{h}$ converges to the dominant eigenvector of the hub matrix $\mathbf{P}\mathbf{P}^T$. Authority vector $\mathbf{a}$ converges to the dominant eigenvector of the authority matrix $\mathbf{P}^T\mathbf{P}$. A survey by Langville et al. [12] has more analysis on the convergence properties of the hub and authority matrices.

4 Topic Sensitivity

Being a global ranking scheme, PageRank has one universal rank vector for all pages on the web. All query results are ranked using this vector. Some broadly popular pages which are heavily inlinked tend to have high pageranks. These pages might be ranked highly for queries in spite of not being relevant because they contain some query terms. PageRank, in its original form, is not "topic-sensitive". HITS, on the other hand, is "topic-sensitive" as it works only on a query-specific subset of the web.

In Section 2 we saw that the aimless surfer model of PageRank allows a low probability random jump from every page to every other page to keep the transition matrix irreducible. This matrix remains irreducible even if this jump is biased towards some pages. The surfer's random jump now takes her to a subset of pages and the ranking of these pages are bound to increase.

Haveliwala's Topic Sensitive PageRank [8] computes a *set* of broad topic biased pagerank vectors offline, and uses them during ranking. There is one PageRank vector per topic. At query time, each topic is assigned a weight based on how "close" the query is to that topic. All topics' PageRank vectors and their query-dependent weights are then used to rank the pages returned by the inverted index. If some additional contextual information regarding the query is known, the weights are based on how close the whole query context is to the topic, instead of just using the query terms. For example: Some query context can be identified if a user highlights certain text on a page and invokes search. User history provides search patterns that can be used to identify context.

A small set of topics C is chosen before hand and for each topic i, a set of web-pages S_i, called *seed* set, is identified. Pages in S_i are chosen such that they have high bearing on topic i. Vector $\mathbf{v_i}$ is constructed from set S_i corresponding to topic i so that:

$$\mathbf{v_i}_j = \begin{cases} \frac{1}{|S_i|}, & \text{if } j \in S_i; \\ 0, & \text{otherwise;} \end{cases} \tag{11}$$

PageRank matrix Equation 3 is modified by introducing $\mathbf{v_i}$ to include the i-th topic specific bias. This is used in conjunction with the eigenvector Equation 4 to compute a topic specific PageRank vector $\mathbf{r_i}$ for each topic i:

$$\mathbf{r_i}^T = \mathbf{r}_i^T \times \left(\alpha(\mathbf{1} \times \mathbf{v_i}^T) + (1-\alpha)\overline{\mathbf{P}}\right) \qquad \forall i \in S_i \tag{12}$$

α is the probability that the aimless surfer jumps to one of the topic specific pages. PageRank typically sets α in between 0.1 and 0.2 to ensure a balance of influence between out-link transitions and random jumps to other pages. Haveliwala chooses $\alpha = 0.25$ so that topic-pages have a greater influence on topic specific PageRank vectors. This process happens offline.

At query time, query-specific weights are given to each of the topics by considering the query context. This is the query dependent part of Topic Sensitive PageRank. In case of regular key word search, the query context is just the query terms. If other contextual information from highlighted search or user history etc. is known, those terms form the context. Let this context be q. Using a multinomial naive-Bayes classifier, with parameters set to their maximum-likelihood, we compute the class probabilities for each of the topics, conditioned on q. Let q_k be the kth term in the query context and for each topic i we compute:

$$P(i|q) = \frac{P(i).P(q|i)}{P(q)} \propto P(i).\prod_k P(q_k|i) \tag{13}$$

Offline, for each topic i, a term vector D_i is constructed so that D_i has all the terms from the seed set of topic i. This is another query independent task that is done along with the PageRank computations. For each possible index term w, $P(w|i)$ is computed for topic i, using its term vector D_i. Now, each term w in the index has i different values corresponding to $P(w|i)$ stored in the index. For the query context term q_k, a simple lookup gives us $P(q_k|i)$. $P(i)$ can be the same for all topics as a typical has no bias towards any topic in C.

Relevant pages for the query are retrieved using an inverted index. Let $\mathbf{r_i}$ denote the PageRank vector corresponding to topic i. The topic sensitive PageRank vector is given by the linear combination of all topic based individual PageRank vectors:

$$\mathbf{r} = \sum_i P(i|q).\ \mathbf{r}_i \tag{14}$$

This method is not that computationally intensive and brings in some amount of query dependence into the ranking process. The number of topics whose PageRank computations are done offline affects the overall computational cost of this method. As the number of eigenvector computations grows linearly with the number of topics, we need to find a balance between the feasible amount of computation and the desired degree of granularity in topic sensitivity. Jeh and Widom [10] propose a dynamic programming model where large overlaps between PageRank vectors are used to increase the number of Topic Sensitive PageRank vectors while incurring significantly less computation cost.

An additional observation is that a user might be biased towards certain topics in C. Some degree of *personalization* can be introduced here by biasing the $P(i)$ distribution so that some topics are more probable for some users. A user's preference of topics can be obtained offline, once, independent of future queries.

5 Stability

The dynamic evolution of the web brings in the problem of rank stability under link perturbation. If a small fraction of hyper-links are changed or removed, ranking of pages should not change drastically. While the notion of quality of a page is still subjective, changes in the dominant eigenvector of the web graph under mild graph perturbations is objective and can be formally studied. Ng et al.[14, 15] study the stability of HITS and PageRank under such graph changes, formally define stability conditions, and suggest improvements to the algorithms that make them more stable.

5.1 Formal Analysis

5.1.1 HITS

Ng et al. prove that HITS is stable under mild perturbations to the web-graph if the *eigengap* of the web-graph is big [14]. The eigengap of a graph is defined as the difference between the first and the second largest eigenvalues. This result is shown by proving that, under mild web-matrix perturbations, these two eigenvalues' magnitudes and their corresponding eigenvectors' directions do not change much. Formally:

Let $\mathbf{A} = P^T P$ be the authority matrix in the HITS process (refer Section 3). Let $\mathbf{a}$ be the dominant eigenvector and δ be the eigengap of $\mathbf{A}$. Assume that the maximum number of out-links out of every web page is bounded by d. For any $\varepsilon > 0$, suppose we perturb the web-graph by adding or deleting at most k links from one page, where $k < (\sqrt{d+\alpha} - \sqrt{d})^2$, where $\alpha = \varepsilon.\ \delta/(4+\sqrt{2\varepsilon})$. Then the perturbed dominant eigenvector $\overline{\mathbf{a}}$ of the perturbed matrix $\overline{\mathbf{A}}$ satisfies $||\mathbf{a} - \overline{\mathbf{a}}||_2 \leq \varepsilon$.

They also prove the converse of the above statement. Suppose $\mathbf{A}$ has an eigengap of δ and $\mathbf{A}$ be perturbed to get $\overline{\mathbf{A}}$ so that $||\mathbf{A} - \overline{\mathbf{A}}||_F = O(\delta)$. This perturbation will cause a large $(\Omega(1))$ change in the dominant eigenvector $\mathbf{a}$.

Proofs of the above are in Ng et al.[14]. HITS is stable w.r.to mild graph perturbations as long as the eigengap is large. If the eigengap is small, mild perturbations to the graph may cause the dominant eigenvector and the secondary eigenvectors to swap places.

5.1.2 PageRank

Ng et al.[14] go on to prove that PageRank is relatively stable under mild graph perturbations. Also, they show that the degree of PageRank perturbation depends on the pageranks of the individual pages that are perturbed. This implies that if low ranked pages are perturbed, the overall PageRank perturbation is quite small.

Formally, let $\mathbf{A}$ be the irreducible aperiodic transition matrix representation of the web-graph (refer Section 2) and $\mathbf{r}$ be its dominant eigenvector. Let pages with pageranks $p_1, p_2, \ldots, p_k$ be changed and $\overline{\mathbf{A}}$ be the new transition matrix. Then the new PageRank vector (dominant eigenvector) $\overline{\mathbf{r}}$ satisfies:

$$||\mathbf{r} - \overline{\mathbf{r}}||_1 \leq \frac{2\sum_{i=1}^{k} p_i}{\varepsilon}$$

It can be seen that the difference in the PageRank vectors is dependent on the pageranks of the perturbed pages. For the intuition behind this statement and its formal proof, refer Ng et al.[14].

This implies that PageRank is reasonably stable under mild perturbations to pages with low pageranks. In a separate result, Haveliwala et al. [9] also prove that PageRank is stable under matrix perturbations. They use the fact that the dominant eigenvector of a transition matrix is stable under perturbations to the underlying Markov chain if its eigengap is large (due to Meyer[13]). If λ_2 is the second eigenvalue of the PageRank matrix, they prove that $\lambda_2 \leq \alpha$ where α is the random jump probability in the aimless surfer model of PageRank (refer Section 2). We know that $0.1 \leq \alpha \leq 0.2$ and if the eigengap is greater than or equal to α, PageRank remains stable.

5.2 Stable Algorithms

In this section we present two algorithms that Ng et al.[15] proposed to stabilize HITS.

5.2.1 Randomized HITS

This model tries to incorporate features of HITS and PageRank into one algorithm. The random jump feature of PageRank is incorporated into the mutual reinforcement of hubs and authorities in the HITS process. The aimless surfer from PageRank has the ability to travel in-links and out-links as if she were following the HITS model. She starts at a random page and on even steps, follows a random in-link of the current page and on odd steps, follows a random out-link of the current page. In both odd and even steps, this current page link (in or out) traversal is done with a probability $1 - \alpha$. With a probability α the surfer jumps to a random page. Thus, the aimless surfer alternates forwards and backwards, and occasionally jumps to a random page.

As the authority and hub matrices under this kind of traversal are similar to the PageRank matrix, their dominant eigenvectors are not vulnerable to mild perturbations. This makes hubs and authorities of Randomized HITS relatively stable.

5.2.2 Subspace HITS

According to matrix perturbation theory, if the eigengap between k-th and $k+1$-th eigenvalues is large, then the subspace spanned by the first k eigenvectors will be stable. The idea here is to consider the first k eigenvectors as the *basis* vectors for a subspace. This subspace, instead of just the dominant eigenvector, is used to obtain authority.

Let the $\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_k}$ be the first k eigenvectors of the authority matrix of HITS ($\mathbf{P}^T\mathbf{P}$) and let $\lambda_1, \lambda_2, \ldots, \lambda_k$ be the corresponding eigenvalues. Let e_j be the j-th basis vector (all 0s except a 1 at the j-th position). The authority scores are given by:

$$a_j = \sum_{i=1}^{k} \phi(\lambda_i)(e_j^T \mathbf{x_i})^2$$

Here $\phi(\lambda_i)$ is a positive monotonically increasing function such as $\phi(\lambda_i) = \lambda_i^2$. Intuitively, each authority score represents the square of the length of the projection of e_j onto the subspace spanned by $x_1, x_2, \ldots, x_k$ where the projection of the x_i direction is weighted by $\phi(\lambda_i)$. This gives a principled way of automatically combining multiple eigenvectors into a single measure of authoritativeness for each page. This method relies on the key result that in general, eigenvector subspaces are more stable than individual eigenvectors.

6 Web Decay

The web has been found to exhibit rapid decay. Links fail, pages go out of date, entire neighborhoods die; the web shows decay as it shows growth. The phenomenon of web decay has been formally analyzed by Bar-Yossef et al. [2]. They use a random walk similar to that of PageRank to compute the "decay" score of a page. As a part of this, they also show that the problem of identifying dead links is not trivial and how dead links are used in the random walk to assess decay.

Though the task of identifying the HTTP 404 return code for a dead link is trivial, the so called *"soft 404 pages"* make the task of identifying logically dead links non-trivial. The soft 404 pages are those that a web server returns in request to some non-existent page from its domain. The returned page might contain some contextual information telling a human user about why the requested page is unavailable. While a human can decipher this message, it is difficult for an automated process to realize that it has hit a soft 404 page. To detect these, the heuristic they use generates known dead links and requests specific web servers with these links. These links are generated by appending random characters to directories inside URLs for domain names served by these web servers. The responses to these known dead links are compared with responses received when unknown URLs are used during crawls. This is one of the ways in which soft 404 pages can be identified. Other heuristics can be found in [2].

We return to the aimless surfer model from PageRank to compute the "decay" score of a page based on a random walk starting from that page. Let N be the total number of pages. Let $D \subset N$

be the set of all dead pages identified during a web crawl. Let P be the adjacency matrix representation of the web-graph such that P_{ij} is the number of links between pages i and j. A self loop is added to each page by $P = P + I$. In the random walk, if page $p \in D$, the surfer returns a decay value of 1 and terminates. If p is not dead, with probability α, it is declared to be live with decay score of zero. With probability $1 - \alpha$, a random out-link from the page is followed and the process is repeated recursively for that page. It is clear that a dead page has decay of 1, a page with dead out-links has decay of $1 - \alpha$, a page with an out-link to a page which has dead links has a decay of $(1 - \alpha)^2$. This measure is tempered by the number of out-links from pages. The formal recursive definition of the decay measure of a page is given by:

$$D_i = \begin{cases} 1, & \text{if } i \in D; \\ (1-\alpha)\left(\frac{\sum_{j \in N} M_{ij} D_j}{\sum_{j \in N} M_{ij}}\right), & \text{otherwise;} \end{cases}$$

The key observation due to the above model is that the decay of a page is independent of the link structure of the entire web, but dependent on the local out-link structure of each page. For the complete random walk model with absorption and experimental results of this model, refer [2].

Eiron et al.[6] suggest four modifications to the basic PageRank algorithm that take decay into consideration.

6.1 Push-Back PageRank

The intuition behind this model is that if a page has a link to a bad page, then it should have its pagerank reduced by a fraction. In turn, this reduced rank part can be "pushed back" to the pages that had given it some of their ranks during the basic PageRank process. A bad page can be a dead link, or some page with high decay. Recalling the basic intuition behind PageRank from Section 2 and taking Equation 2 is taken in the converging sense, we have:

$$\mathbf{r}^T = \mathbf{r}^T \mathbf{P} \tag{15}$$

Eiron et al.[6] suggest that a page's penalized rank should be returned to its contributors in the same proportion as the rank was bestowed. The penalized page i should retain a proportion $(1 - \beta_i)$ of its original rank and the remaining rank should be distributed in proportion $P_{ji}\beta_i$ to all the contributing pages j. In matrix form this can be written as:

$$\mathbf{r}^T = \mathbf{r}^T \mathbf{P} \mathbf{B} \tag{16}$$

If the first page is being penalized, then B will look like:

$$\mathbf{B} = \begin{pmatrix} (1-\beta_i) & \beta \mathbf{p_1} \\ 0 & I \end{pmatrix}$$

The rows are normalized to 1 so that row stochastic property is retained. $\mathbf{p_1}$ is the first row of $\mathbf{P}$ without p_{11}. In the event that several pages are penalized, this matrix can be extended so that all the penalized pages push back a fraction $(1 - \beta_i)$ of their rank to their in-links. β_i can be chosen to be the fraction of dead links in a page to the total number of links from the page. The decay factor of a page as computed by Bar-Yossef et al.[2] can be used to determine β_i. In this report we have used the row stochastic representation of the matrices while Eiron et al.[6] use the column stochastic approach. The above approach incurs an extra computation step in the PageRank process. As web decay increases, this extra step of computation should improve ranking considerably.

6.2 Self-Loop approach

In this approach, self loops are added to the transition matrix so that some amount of a page's rank is kept to itself. During the PageRank iterations, the aimless surfer at page i takes the self loop with some probability γ_i. For pages with high decay, γ_i is kept low and pages with low decay get high γ_i so that a good page accumulates more pagerank due to its self loops.

6.3 Jump weighting approach

Recall from Section 2 that in PageRank, with probability α, the aimless surfer jumps to a random page out of all pages. The selection of this random page can be restricted to only pages with a decay measure below some threshold. This will ensure that high decay pages don't get these random jumps and hence get low ranks.

6.4 BHITS

An in-link reinforcement akin to HITS is added to PageRank in this approach. This model also addresses the problem of dangling links (pages with no out-links). A dangling link might be a dead page, or a page with information but no out-link. Recall from Section 2 that in the PageRank model a dangling node is either removed or an equally probable jump to any page is allowed. In the BHITS random walk of the web-graph, if we encounter a dangling link, we can treat it in one of two ways: 1 - If it is a dead link, the rank that was supposed to be given to it is distributed randomly across the web (the aimless surfer takes a random jump instead of visiting the node). 2 - If the link is just dangling, but has information, pagerank from here is not distributed to all pages (like classic PageRank), but it is distributed to its in-links by a backward traversal step. Each in-link gets an equal share of the dangling page's rank. This backward step is done only for pages with no out-links. For normal pages, a backward traversal means only a self loop. That is, normal pages do not lose any ranking during the backward step. This scheme reduces the ranks of dead links and pages with high decay, but preserves ranking of pages which point to zero-out-link pages.

7 Fine Grained Page Analysis

Ranking methods like PageRank and HITS, in their original sense, treat pages at a macroscopic level. This *coarse grained* model of the web is getting obsolete as web pages are getting more complex. This is due to pages evolving to posses rich, structured, semi-structured and other complex layouts often including banners, navigation panels, advertising sections etc. Ranking methods like PageRank and HITS, which do not distinguish between "relevant" and "irrelevant" links, tend to diffuse rank from a page to all its out-links. The task, therefore, is to analyze pages in *fine grained* ways so that relevant and irrelevant links are identified. This problem of analyzing sub-pages during ranking needs to be addressed differently for PageRank and HITS. As PageRank inherently does not have a context (due to its query independence), every link on a web page might be relevant during ranking. Before we conclude this report, we will discuss ways in which sub-page analysis might contribute to PageRank. First, we study DomTextHits, an improvement over HITS that decomposes hubs into context specific *micro-hubs* during ranking. This is due to Chakrabarti et al. [5, 3].

7.1 DomTextHits

As the HITS ranking algorithm proceeds, hub and authority scores of each page in the base set are reinforced mutually based on hyperlinks between them (refer Section 3). The key observation made here is that current web-pages that form the base set contain the so called "mixed-hubs". These hubs tend to have links to authorities on more than one topic. HITS expects each hub to point to authorities that are relevant only to the given query and diffuses rank to these authorities. HITS does not perform well with mixed-hubs because "query-specific" rank will get diffused into query irrelevant authorities that are linked by the mixed-hub. This is referred to as *topic drift.* To avoid topic drift, Chakrabarti et al. suggest that each hub be treated as a Document Object Model (DOM) tree. In this DOM tree, all "micro-hubs" pointing to pages that are from a single authority topic form DOM subtrees. These subtrees can be identified by using just information theoretic models (DomHits: see [3]) or DomHits coupled with textual analysis of pages (DomTextHits: see [5]).

In both these models, hub pages are represented using DOM trees and DOM *subtree* roots are considered as sub-pages. This hierarchical structure of each page is due to its mixed-hub nature. Each micro-hub under the document root has other micro-hubs under it and so on. Eventually, leaf level nodes have hyperlinks to the authorities but importantly, query-pertinent authorities come under only a subset of these micro-hubs at some level of hierarchy of the DOM tree. DomTextHits identifies these query-specific micro-hubs in a page's DOM tree based on two key observations: 1 - query-specific micro-hubs tend to have clustered links to query-specific authorities. 2 - query-specific micro-hubs have more query-specific textual content in them.

The first point appears to be the definition of query-specific micro hubs. Earlier, we were under the impression that DomTextHits would discover these query specific micro-hubs based on some other intelligence. But now, somewhat counter-intuitively the algorithm is looking for them based on how they are defined. This works due to the following modification to HITS. Each DOM tree root (every page) and all its subtrees are considered as a node in the constructed web graph. Note that there are more nodes in the graph now as compared to that of plain HITS. The algorithm initializes unit authority scores to only those nodes which correspond to DOM tree roots of root set documents. The score reinforcement operation $\mathbf{h}^{t+1} = \mathbf{P} \times \mathbf{a}^t$ then transfers these initial authority scores to all hub leaves which link to them. Now, recall that only query-specific authorities have been selected to form the base set and after the authority $\rightarrow$ hub score transfer, only relevant hub leaves will have hub scores. The irrelevant hub leaves which have non-query specific links do not get any hub scores. The hub subtrees which have the best such score (based on their leaves' scores) need to be chosen as the query-specific micro-hubs. The leaf scores of these subtrees are collected and distributed back to them so that all micro-hub leaves have the same score. The $\mathbf{a}^{t+1} = \mathbf{P}^T \times \mathbf{h}^t$ transfer happens after this step. We notice that now, this step does not “leak” hub scores to irrelevant authorities. To implement this idea, the frontier where leaf hub scores can be summed up for redistribution has to be found. An information theoretic concept called Minimum Description Length (MDL) [17] is used to do this. For details, see [5, 3, 4].

Query specific micro-hubs can also be identified by using textual content in them. The standard TFIDF-weighted vector space centroid (see [4]) of all the root set documents gives us the text-term distribution of the query; [5] calls this the ground truth vector. For each subtree in a DOM tree, the IDF scaled vector of text in the subtree is compared (on cosine similarity) against the ground truth vector. If the similarity is large enough, this subtree root is chosen as the node till which hub scores are added and redistributed back. If the similarity is not large enough (implying that there might be mixed hubs underneath this node), the next layer of subtrees is considered and so on.

A combination of the above two parameters is used in DomTextHits to avoid topic drift and get better topic distillation. For performance metrics of HITS, DomTextHits and DomHits, and other details, refer to [5, 3, 4].

8 Link Spamming

In the link-analysis context, spamming refers to spurious (and often commercially driven) hyperlinking between pages that are not endorsed by any editorial judgment regarding relevance. Combating link-spam has become an interesting[2] contest between search engines and commercially driven search engine optimization companies. We briefly review how certain aspects of PageRank and DomTextHits help combat link-spam.

PageRank can be manually tweaked to combat link-spam. This is done by ensuring that the aimless surfer does nor randomly jump to link-spam neighborhoods. This is done quite easily by lowering their probabilities in the bias vector. However, this approach needs *a priori* knowledge of link-spam neighborhoods. Automating a part of this process is explored in TrustRank [7]. DomTextHits inherently combats link-spam due to its query-specific discarding of irrelevant subtrees (spam).

9 Further Research

Web Search still remains far from satisfactory. There still remains a large unbridged gap between the query thought in a user's mind and the results that are displayed to her. Ongoing and further research in this space can be roughly categorized into five sections. 1 - Index coverage; 2 - Intent driven retrieval; 3 - Ranking; 4 - Scale; 5 - Other. Even if research in other domains brings in improvement, research in ranking of results is still critical to Web Search due to a typical user's short attention time and impatience.

Also, PageRank, HITS and their extensions/modifications lack rigorous theoretical foundations for their guarantee on relevance. Achlioptas et al. [1] have given a theoretically sound model for ranking. This model is empirically not as well tested as PageRank and HITS. Bringing well tested models like PageRank and HITS closer to rigorous theoretical models like [1] and vice versa is another area of further research. The importance vs. relevance question that arises in query-independent ranking schemes like PageRank also needs to be explored further. Personalizing the ranking for individual users also brings in the engineering aspects required to handle such scale.

Any further research, unless fully backed by formal rigorous theory, needs empirical testing to validate effectiveness. Search engines like Google and Yahoo!, with large testing infrastructure and millions of real life test subjects, have a scale advantage during testing. Academic research in this area is either forced to borrow testing infrastructure or extrapolate based on small world models. Chakrabarti et al. [5] propose a technique to test ranking objectively by using an open directory like DMoz. Overall, using real test subjects still seems to be most effective as they are the eventual consumers, and their behavior has not been modeled objectively yet.

10 Conclusion

In this report, after reviewing eigenvector based ranking methods, we saw various other parameters that effect ranking. With the continuous evolution of the Web we can expect many more of such parameters to emerge in the future. With the Web becoming widely accepted as an authoritative source for all kinds of information, there is a pressing need for research in the ranking space, eventually helping search engines serve users better.

References

[1] Dimitris Achlioptas, Amos Fiat, Anna R. Karlin, and Frank McSherry. Web search via hub

[2]for the academic, of course

synthesis. In *IEEE Symposium on Foundations of Computer Science*, pages 500–509, 2001.

[2] Ziv Bar-Yossef, Andrei Z. Broder, Ravi Kumar, and Andrew Tomkins. Sic Transit Gloria Telae: Towards an Understanding of the Web's Decay. In *Proceedings of the 13th International World Wide Web Conference*, 2004.

[3] Soumen Chakrabarti. Integrating the Document Object Model with Hyperlinks for Enhanced Topic Distillation and Information Extraction. In *Proceedings of the 10th International World Wide Web Conference*, 2001.

[4] Soumen Chakrabarti. *Mining the Web: Discovering Knowledge from Hypertext Data*. Morgan-Kauffman, 2002.

[5] Soumen Chakrabarti, Mukul M. Joshi, and Vivek B. Tawde. Enhanced Topic Distillation using Text, Markup Tags and Hyperlinks. In *Proceedings of SIGIR*, 2001.

[6] Nadav Eiron, Kevin S. McCurley, and John A. Tomlin. Ranking the Web Frontier. In *Proceedings of the 13th International World Wide Web Conference*, 2004.

[7] Zoltan Gyöngyi, Hector Garcia-Molina, and Jan Pederson. Combating Web Spam with TrustRank. In *Proceedings of the 30th International Conference on Very Large Data Bases*, 2004.

[8] Taher Haveliwala. Topic-Sensitive PageRank: A Context-Sensitive Ranking Algorithm for Web Search. In *IEEE Transactions on Knowledge and Data Engineering*, 2003.

[9] Taher Haveliwala and S. Kamvar. The Second Eigenvalue of the Google Matrix. Technical report, Stanford University, 2003.

[10] Glen Jeh and Jennifer Widom. Scaling Personalized Web Search. In *Proceedings of the Twelfth World Wide Web Conference*, 2003.

[11] Jon Kleinberg. Authoritative Sources in a Hyperlinked Environment. *Journal of the ACM*, 1999.

[12] A. N. Langville and C. D. Meyer. A Survey of Eigenvector Methods of Web Information Retrieval. In *SIAM Review (Forthcoming)*.

[13] C. D. Meyer. Sensitivity of the Stationary Distribution of a Markov Chain. *SIAM Journal on Matrix Analysis and Applications*, 1994.

[14] Andrew Y. Ng, Alice X. Zheng, and Michael I. Jordan. Link Analysis, Eigenvectors and Stability. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence*, 2001.

[15] Andrew Y. Ng, Alice X. Zheng, and Michael I. Jordan. Stable Algorithms for Link Analysis. In *Proceedings of SIGIR*, 2001.

[16] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank Citation Ranking: Bringing Order to the Web. Technical report, Stanford Digital Library Technologies Project, 1998.

[17] J. Rissanen. Modelling by Shortest Data Description. *Automatica*, 14:465–471, 1978.