International Journal of Modern Engineering Research (IJMER)
www.ijmer.com Vol.2, Issue.1, Jan-Feb 2012 pp-375-383 ISSN: 2249-6645

ANALYSIS OF DATA MINING TECHNIQUES FOR INCREASING SEARCH SPEED IN WEB

B.Chaitanya Krishna[1],C.Niveditha[2],G.Anusha [2],U.Sindhu [2],Sk.Silar[2]
1(Assistant Professor, Dept. of Computer Science And Engineering, KL University.)
2(B.Tech Scholars, Dept. of Computer Science And Engineering, KL University.)

ABSTRACT
The World Wide Web contains an enormous amount of information, but it can be exceedingly difficult for users to locate resources that are both high in quality and relevant to their information needs. Issues that have to be dealt with are the detection of relevant information, involving the searching and indexing of the Web content, the creation of some metaknowledge out of the information which is available on the Web, as well as the addressing of the individual users' needs and interests, by personalizing the provided information and services. In this paper we discuss mainly two algorithms which increase the search engine speed.

Keywords: World Wide Web, Search Engines, Information Retrieval, PageRank, Google,HITS.

I.INTRODUCTION
Data mining nowadays plays an important role in searching the information on the web that include a high variety data types. For reaching this goal, datamining techniques for automatic discovering and extracting the web based information has been used as webmining. In this article, data mining which is a new method in retrieving the high amount of information has been introduced.

Every day, the WWW grows by roughly a million electronic pages, adding to the hundreds of millions already on-line. Because of its rapid and chaotic growth, the resulting network of information lacks of organization and structure. Moreover, the content is published in various diverse formats. Due to this fact, users are feeling sometimes disoriented, lost in that information overload that continues to expand.

II.PAGERANK ALGORITHM
2.1.Bringing Order to the Web
The citation (link) graph of the web is an important resource that has largely gone unused in existing web search engines. Maps are created containing as many as 518 million of these hyperlinks, a significant sample of the total. These maps allow rapid calculation of a web page's "PageRank", an objective measure of its citation importance that corresponds well with people's subjective idea of importance. Because of this correspondence, PageRank is an excellent way to prioritize the results of web keyword searches. For most popular subjects, a simple text matching search that is restricted to web page titles performs admirably when PageRank prioritizes the results (demo available at google.stanford.edu). For the type of full text searches in the main Google system, PageRank also helps a great deal.

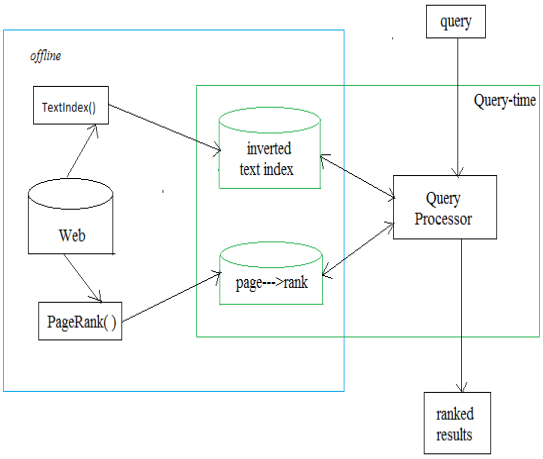

Figure 1: Simplified diagram illustrating a simple search engine utilizing the standard PageRank

Algorithm:Improved PageRank(G,s,k)

1: d <= 0.85
2: n <=number of vertices of G
3: for i = 0 to n do
4: pr[i] <= s
5: end for
6: for all E(G) do
7: t <=source(e)
8: out[i] = out[i] + 1
9: end for
10: for j = 0 to k do

11: for all E(G) = e do
12: (t,h) <= (source(e), target(e))
13: $pr_{in}[h] = pr_{in}[h] + pr[t]/out[t]$
14: end for
15: end for
16: for i = 0 to n do
17: pr[i] = (1- d) + d(pr_{in}[i])
18: end for
19: avg <= $\frac{sum(pr[i])}{n}$
This implementation has a complexity of
O(k m + n)

where k =number of iterations,
G=graph
s=integer value
n =|V (G)|, m = |E(G)|

2.2.Description of PageRank Calculation

Academic citation literature has been applied to the web, largely by counting citations or back links to a given page. This gives some approximation of a page's importance or quality. PageRank extends this idea by not counting links from all pages equally, and by normalizing by the number of links on a page. PageRank is defined as follows:

We assume page A has pages T1...Tn which point to it (i.e., are citations).The parameter d is a damping factor which can be set between 0 and 1.We usually set d to 0.85. There are more details about d in the next section. Also C(A) is defined as the number of links going out of page A. The PageRank of a page A is given as follows:

PR(A) = (1-d) + d (PR(T1)/C(T1) + ... + PR(Tn)/C(Tn))

Note that the PageRanks form a probability distribution over web pages, so the sum of all web pages' PageRanks will be one. PageRank or PR(A) can be calculated using a simple iterative algorithm, and corresponds to the principal eigenvector of the normalized link matrix of the web. Also, a PageRank for 26 million web pages can be computed in a few hours on a medium size workstation.

2.3.Google Architecture Overview

Most of Google is implemented in C or C++ for efficiency and can run in either Solaris or Linux. In Google, the web crawling (downloading of web pages) is done by several distributed Crawlers. There is a URL server that sends lists of URLs to be fetched to the crawlers. The web pages that are fetched are then sent to the store server. The store server then compresses and stores the web pages into a repository. Every web page has an associated ID number called a doc ID which is assigned whenever a new URL is parsed out of a web page. The indexing function is performed by the indexer and\ the sorter. The indexer performs a number of functions. It reads the repository, uncompresses the documents, and parses them. Each document is converted into a set of word occurrences called hits. The hits record the word, position in document, an approximation of font size, and capitalization. The indexer distributes these hits into a set of "barrels", creating a partially sorted forward index. The indexer performs another important function. It parses out all the links in every web page and stores important information about them in an anchors file. This file contains enough information to determine where each link points from and to, and the text of the link. The URL resolver reads the anchors file and converts relative URLs into absolute URLs and in turn into doc IDs. It puts the anchor text into the forward index, associated with the doc ID that the anchor points to. It also generates a database of links which are pairs of doc IDs. The links database is used to compute Page Ranks for all the documents.

The URL resolver reads the anchors file and converts relative URLs into absolute URLs and in turn into docIDs. It puts the anchor text into the forward index, associated with the docID that the anchor points to. It also generates a database of links which are pairs of docIDs. The links database is used to compute PageRanks for all the documents. The sorter takes the barrels, which are sorted by docID[1] and resorts them by wordID to generate the inverted index. This is done in place so that little temporary space is needed for this operation. The sorter also produces a list of wordIDs and offsets into the inverted index. A program called DumpLexicon takes this list together with the lexicon produced by the indexer and generates a new lexicon to be used by the searcher. The searcher is run by a web server and uses the lexicon built by DumpLexicon together with the inverted index and the PageRanks to answer queries.

2.4.Google Query Evaluation

1. Parse the query.
2. Convert words into wordIDs.
3. Seek to the start of the doclist in the short barrel for every word.
4. Scan through the doclists until there is a document that matches all the search terms.
5. Compute the rank of that document for the query.
6. If we are in the short barrels and at the end of any doclist, seek to the start of the doclist in the full barrel for every word and go to step 4.

7. If we are not at the end of any doclist g o to step 4. Sort the documents that have matched by rank and return the top .

2.5.Limitations of pageRank

As we know that in PageRank the web pages are ranked according to the number of clicks made on that particular web page but this may lead to illegal ranking of web pages i.e.,whenever a query is given the pages that are satisfying the query are presented according to the rank of the page. The top most one will be given highest priority. The highest priority is because the number of clicks on that particular web page are more without concerned with the content that is present in that particular web page. For this purpose the ranking should be given according to the content present in the web page rather than the number of clicks made on that particular web page. Because if a wrong page is presented to end user then he will browse the page which will increase the click count of the traced page which is wrong. This leaves the web page with highest priority. This will continue furthurly. For this purpose it is better to rank pages according to the content in the web page. This leads to the combination of text mining with web mining.

III. HITS ALGORITHM

3.1. An overview of the HITS algorithm

HITS (Hyperlink-Induced Topic Search**)** algorithm mines the link structure of the Web and discovers the thematically related Web communities that consist of 'authorities' and 'hubs.' Authorities are the central Web pages in the context of particular query topics. For a wide range of topics, the strongest authorities consciously do not link to one another. Thus, they can only be connected by an intermediate layer of relatively anonymous hub pages, which link in a correlated way to a thematically related set of authorities [2].

A good hub page for a subject points to many authoritative pages on that content, and a good authority page is pointed by many good hub pages on the same subject. We should stress that a page might be a good hub and a good authority in the same time. This circular relationship leads to the definition of an iterative algorithm,HITS.

These two types of Web pages are extracted by iteration that consists of following two operations.

$$x_p = \sum_{q,q \to p} y_p$$

$$y_p = \sum_{q,p \to q} x_q$$

For a page p the weight of x_p is updated to be the sum of y_p over all pages q that link to p: where the notation $q->p$ indicates that q links to p In a strictly dual fashion, the weight of y_p is updated to be to the sum of x_q. Therefore, authorities and hubs exhibit what could be called mutually reinforcing relationships: a good hub points to many good authorities, and a good authority is pointed to by many good hubs.

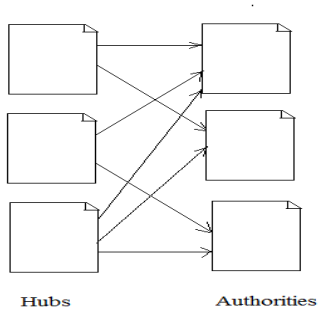

Figure 2: Illustration of Hub and Authorities

The whole picture of HITS algorithm is shown as follows:

step 1: Collect the r highest-ranked pages for the query σ from a text-based search engine such as AltaVista[3]. These r pages are referred as the root set Rσ.

step 2: Obtain the base set $S\sigma$ whose size is n by expanding $R\sigma$ to include any page pointed to by pages in $R\sigma$ and at most d pages pointing to pages in $R\sigma$.

step 3: Let G[$S\sigma$] denote the subgraph induced on the pages in $S\sigma$. Two types of links in G [$S\sigma$] are distinguished as *transverse links* and *intrinsic links*. The former are the links between pages with different domain names, and the latter
are the ones between pages with the same domain name. All *intrinsic links* from the graph $S\sigma$ are deleted, keeping only the edges corresponding to *transverse links*.

step 4: Make the n by n adjacency matrix A and its transposed matrix A^T. Normalized principal eigenvector e_1 of $A_T$that corresponds to the largest eigenvalue λ_1 is obtained
by eigenvalue calculation.

step 5: Find elements with large absolute values in the normalized principal eigenvector e_1. Return them as 'authorities.'

3.2.Pseudocode for HITS algorithm

```
1  G := set of pages
2  for each page p in G do
3  p.auth = 1
 // p.auth is the authority score of the page p
4  p.hub = 1 // p.hub is the hub score of the page p
5  function HubsAndAuthorities(G)
6  for step from 1 to k do
 // run the algorithm for k steps
7  norm = 0
8  for each page p in G do
 // update all authority values first
9  p.auth = 0
10  for each page q in p.incomingNeighbors do  //
p.incomingNeighbors is the set of pages that link  to
p
11  p.auth += q.hub
12  norm += square(p.auth)
// calculate the sum of the squared auth values to
normalize
13  norm = sqrt(norm)
14  for each page p in G do  // update the auth scores
15  p.auth = p.auth / norm  // normalize the auth
values
16  norm = 0
17  for each page p in G do  // then update all hub
values
18  p.hub = 0
19  for each page r in p.outgoingNeighbors do
//p.outgoingNeighbors is the set of pages that p links
to
20  p.hub += r.auth
21  norm += square(p.hub) // calculate the sum of
the squared hub values to normalize
22  norm = sqrt(norm)
23  for each page p in G do  // then update all hub
values
24  p.hub = p.hub / norm  // normalize the hub
values
```

3.3. Problems with the HITS Algorithm

To clarify problems with HITS algorithm, we traced Kleinberg's experiments. We picked 9 query topics for our study: 'abortion,' 'Artificial Intelligence,' 'censorship,' 'Harvard,' 'jaguar,' 'Kyoto University,' 'Olympic,' 'search engine,' and 'Toyota.' In these query topics, all but 'Kyoto University' and 'Toyota' were used in [2] and [3]. Though we fixed the parameters , , and a text-based search engine for collecting the root set to examine Kleinberg's experiments rigorously, we observed HITS algorithm performed poorly in several of our test cases. In this paper, we discuss focusing on topic 'Artificial Intelligence' as a successful example, and topic 'Harvard'as an unsuccessful example.

Topic: 'Artificial Intelligence'

The extracted top 5 authorities and hubs of 'Artificial Intelligence' in our experiment are indicated in Table 1. The decimal fractions shown on the left of URLs represent authority weights () and hub weights () respectively The top authority was the home page of JAIR (Journal of Artificial Intelligence Research), the second authority was AAAI (American Association for Artificial Intelligence), then MIT AI laboratory followed. Namely, famous organizations related to Artificial Intelligence based in the United States were successfully extracted. This AI community was supplemented by hubs, which consisted of the researcher's personal Web pages (e.g. S. Russell at UCB)

x_p	**Authorities**
.372	http://www.cs.washington.edu/research/jair/home.html
.298	http://www.aaai.org/
.294	http://www.ai.mit.edu/
.272	http://ai.iit.nrc.ca/ai_point.html
.234	http://sigart.acm.org/

y_p	**Hubs**
.228	http://yonezaki- www.cs.titech.ac/jp/member/hidekazu/Work/AI.html
.228	http://www.cs.berkley.edu/~russell/ai.html
.204	http://uscial.usu.clu.edu/pantonio/cco360/AIWeb.htm
.181	http://www.scms.rgu.ac.uk.staff/asga/ai/html
.171	Tpp://www.ex.ac.uk/ESE/IT/ai.html

(note: x_p and $y_p$represent authority weight and hub weight respectively)

Table 1: Authorities and hubs of 'Artificial Intelligence.'

Topic: 'Harvard'

In Kleinberg's experiment, authorities of 'Harvard' were related to Harvard University; e.g. the homepage of Harvard University, Harvard Law School, Harvard Business School, and so on. However, in our experiment, the Web pages authored by a financial consulting company were extracted (see Table 2). These pages did not relate to query 'Harvard.'

x_p	**Authorities**
.130	http://www.wetradefutures.com/investment.asp
.130	http://www.wetradefutures.com/trend.htm
.130	http://www.wetradefutures.com/market_technology.htm
.130	http://www.wetradefutures.com/florida_investment.htm
.130	http://www.wetradefutures.com/investing_investment.htm
y_p	**Hubs**
.247	http://www.profittaker.net/data.htm
.247	http://www.profittaker.org/new_twentyseven.htm
.247	http://www.profittaker.com/sunday_trader_more.htm
.247	http://www.profittaker.cc/system_software.htm
.247	http://www.profittaker.com/contact_phone.htm

(note: x_p and y_p represent authority weight and hub weight respectively)

Table 2: Authorities and hubs of 'Harvard.'

In this case, higher ranked 56 authorities had the same authority weights, and higher ranked 5 hubs had the same hub weights. By checking the contents of these pages, we detected that these authorities and hubs were authored by a single organization.

In the HITS algorithm, the first step is to retrieve the set of results to the search query. The computation is performed only on this result set, not across all Web pages.

Authority and hub values are defined in terms of one another in a mutual recursion. An authority value is computed as the sum of the scaled hub values that point to that page. A hub value is the sum of the scaled authority values of the pages it points to. Some implementations also consider the relevance of the linked pages.

The algorithm performs a series of iterations, each consisting of two basic steps:

- **Authority Update**: Update each node's Authority score to be equal to the sum of the Hub Scores of each node that points to it. That is, a node is given a high authority score by being linked to by pages that are recognized as Hubs for information.
- **Hub Update**: Update each node's Hub Score to be equal to the sum of the Authority Scores of each node that it points to. That is, a node is given a high hub score by linking to nodes that are considered to be authorities on the subject.

The Hub score and Authority score for a node is calculated with the following algorithm:

- Start with each node having a hub score and authority score of 1.
- Run the Authority Update Rule
- Run the Hub Update Rule
- Normalize the values by dividing each Hub score by the sum of the squares of all Hub scores, and dividing each Authority score by the sum of the squares of all Authority scores.
- Repeat from the second step as necessary.

HITS, like PageRank and Brin's PageRank, is an iterative algorithm based on the linkage of the documents on the web. However it does have some major differences:

- It is executed at query time, not at indexing time, with the associated hit on performance

that accompanies query-time processing. Thus, the *hub* and *authority* scores assigned to a page are query-specific.

- It is not commonly used by search engines.
- It computes two scores per document, hub and authority, as opposed to a single score.
- It is processed on a small subset of 'relevant' documents, not all documents as was the case with PageRank.

3.3.1 In Detail

To begin the ranking, $\forall p$, auth(p) = 1 and hub(p) = 1. We consider two types of updates: Authority Update Rule and Hub Update Rule. In order to calculate the hub/authority scores of each node, repeated iterations of the Authority Update Rule and the Hub Update Rule are applied. A k-step application of the Hub-Authority algorithm entails applying for k times first the Authority Update Rule and then the Hub Update Rule.

3.3.2.Authority Update Rule

For all p, we update auth(p) to be: $\sum_{i=1}^{n} hub(i)$ where n is the total number of pages connected to p and i is a page connected to p. That is, the Authority score of a page is the sum of all the Hub scores of pages that point to it.

3.3.3.Hub Update Rule

For all p, we update hub(p) to be: $\sum_{i=1}^{n} auth(i)$ where n is the total number of pages p connects to and i is a page which p connects to. Thus a page's Hub score is the sum of the Authority scores of all its linking pages

3.3.4.Normalization

The final hub-authority scores of nodes are determined after infinite repetitions of the algorithm. As directly and iteratively applying the Hub Update Rule and Authority Update Rule leads to diverging values, it is necessary to normalize the matrix after every iteration. Thus the values obtained from this process will eventually converge.

3.4.Further problems with the HITS algorithm:

The HITS algorithm does not always behave as expected. First, if the dominant eigenvalue of M^TM is repeated, the HITS algorithm converges to an authority vector which is *not unique*, but depends on the initial seed $\overrightarrow{a_0}$.The authority vector can be any normalized vector in the dominant eigenvalue's eigenspace. For example, for a two-level reversed binary tree B whose edges point upwards towards the root, the eigenvalues of M^TM are 2, 2, 2, 0, 0, 0, and 0. The authority weights for the three upper nodes can be any three positive numbers that sum to 1. Second, the HITS algorithm yields *zero* authority weights for apparently important nodes of certain graphs. For example, if a leaf is added at the left middle-level node of B, then both the hub and authority weights are zero for the root and for the right half of B. We call these limitations *non-uniqueness* and *nil-weighting*, respectively.

The graphs G that are characterized are those on which the HITS algorithm is non-unique or nil-weighted. Consider an undirected graph G^1on [n] where { i,j} is an edge of G^1if there is a k such that (k,i) and (k,j) are directed edges of G. The HITS algorithm is non-unique or nil-weighted on G if and only if there exist i,j with positive in-degree in G such that i and j are in distinct components of G^1.

A disadvantage of the algorithm is that because it is executed at query time, it may have very poor performance, depending on the number of iterations required to rank the Hubs and authorities.

A second disadvantage of the algorithm is that it may be making assumptions about the structure of the Web that no longer hold true. In the early days, there were many Web pages that were legitimate collections of links. This was due to the poor quality of search results. With the advent of better search engine technology these collections have assumed a minor role, and have generally resolved themselves into

1- Legitimate human edited directories like www.dmoz.org

2- Link farms and directories for exploitation purposes.

3- "Link bait" pages that have useful links in order to draw links from others.

4- Pages that are themselves authoritative, but also include a number of external links, such as Wikipedia.

5- Link pages based on mutual link exchanges. Like "link bait" collections, these can be legitimate sources of authority and relevance in some cases. Political action groups are going to exchange links with like-minded groups.

The algorithm doesn't seem to take account of case number 4 - "hubs" that are also authorities, and

which are perhaps the best indicators of good external pages. On the other hand, results might be unduly influenced by link farms. It might be able to do away with the tricky identification of hubs and iterative ranking simply by relying on a number of good directories like the dmoz open directory project.

IV. COMPARATIVE ANALYIS

Algorithm	PageRank	HITS
Main Technique	Web Strucutre Mining	Web Structure Mining,Web Content Mining
Methodology	This algorithm computes the score for pages at the time of indexing of the pages.	It computes hubs and authority of the relevant pages.
Input Parameter	Black links	Content,Back,forward links
Relevancy	Less(this algorithm rank the pages on the indexing time)	More (this algorithm uses the hyper links so according to Henzinger,2001 it will give good results and also consider the content of the page)
Quality of results	Medium	Less than PR
Importance	High. Back links are considered.	Moderate. Hubs and authorities scores are utilized.
Limitation	Results come at the time of indexing and not at the query time.	Topic drift and efficiency problem.

Table 3: Summary of web page ranking algorithms

V. TOP 10 SEARCH ENGINE OPTIMIZATION TECHNIQUES

There are a lot of techniques that can be used in search engine optimization that can help or hurt you when it comes to SEO. Some of the big ones that can hurt is keyword stuffing and page cloaking. You want to always make sure that you are building a site for your visitors first and foremost. The robots aren't quite as interested in your content as you think, sorry.

You can easily make your site search engine optimized by follow 10 easy steps that will make sure everything is ready for Google to go through and understand what your pages and site are about. This can help you to get more traffic from search engines by ranking well. Once you rank well, of course, you need to make sure that you are persuading the visitors to actually click on your search listing.

5.1.Content

This is the number one for any search marketing strategy, it is impossibly important to ensure that you have content worth viewing. Without this one simply step to ensure that there is a reason for someone to be on your site, everything else is useless. There are a lot of great sites to find inspiration for writing great content that works.

5.2.Incoming Links

A link is a link is a link, but without the simplest form you aren't going to do well in search engines. The more links you have the more often you are going to be crawled. It is also important to make sure that you have the proper anchor text for your incoming links. The easiest way to gain quality links from other sites is to link to sites to let them know your site is there and hope for a reciprocal link. It is also important to make sure that you have content that is worth linking to on your site.

5.3. Web site title

Making sure that you have the right web site titles for your pages is extremely important. The keywords you place in your title are important in order to ensure that your topic is understood by Google. One of the primary factors for ranking is if the title is on-topic with the search results. Not only is it important for robots to index and understand the topic of the page either. It is important for click-through rates in the search results. Pay attention to what you click on when you are searching in Google, I know that I don't always click the first results. Using great titles and topics on your site will bring you more traffic than a number one listing. Most of the time it is within the first page, but I skim through the titles to see which looks to be more on-topic for my search query.

5.4.Heading tags

When you are laying out your site's content you have to be sure that you are creating the content flow in such a way that the heading tags are based on prominence. The most prominent of course being the h1 tag, which says â€œthis is what this block of copy is about.â€• Making sure you understand heading

tag structure is very important. You only want to have one (or two) h1 tags per a page. It is important to not just throw anything into an h1 tag and hope you rank for it.

5.5.Internal Linking

Making sure that your internal linking helps robots (and visitors!) to find the content on your site is huge. Using relevant copy throughout your site will tell the robots (and visitors!) more effectively what to expect on the corresponding page. You do want to make sure that on pages you don't want to rank in Google that you add a nofollow tag to ensure that the ranking flow of your site corresponds with your site's topic and interests. No one is going to be searching Google to find out what your terms of service or privacy policy are.

5.6. Keyword Density

Ensuring that you have the right keyword density for your page and sites topic is paramount. You don't want to go overboard and use the keyword every 5th word but making sure it â€œcomes upâ€• often is going to help you rank better in search engines. The unspoken rule is no more then 5% of the total copy per a page. Anymore then this and it can start to look a little spammy. Granted, you aren't shooting for 5% every time. It is really all about context and relevanceâ€" just make sure it is good, quality copy.

5.7. Sitemaps

It is always a good idea to give search engines a helping hand to find the content that is on your site. Making sure that you create and maintain a sitemap for all of the pages on your site will help the search robots to find all of the pages in your site and index them. Google, Yahoo, MSN and Ask all support sitemaps and most of them offer a great way to ensure that it is finding your sitemap. Most of the time you can simply name it sitemap.xml and the search robot will find the file effectively.

5.8.Meta Tags

Everyone will tell you that meta tags don't matter, they do. The biggest thing they matter for is click-through though. There will be a lot of times when Google will use your meta description as the copy that gets pulled with your search listing. This can help to attract the visitor to visit your web site if it is related to their search query. Definitely a much overlooked (as of late) ranking factor. Getting indexed by search engines and ranking well is just the first step. The next, and biggest, step is getting that visitor that searched for your keywords to want to click on your search listing.

5.9.URL Structure

Ensuring that your URL structure compliments the content that is on the corresponding page is pretty important. There are various methods to make this work, such as modrewrite on apache.

5.10.Domain

It can help to have keywords you are interested in ranking for within your domain, but only as much as the title, heading and content matters. One very important factor that is coming to light is that domain age is important. The older the site or domain, the better it is not spam and can do well in search results. The domain age definitely isn't a make or break factor but it does help quite a bit.

V.CONCLUSION

The www has grown into a hypertext environment of enormous complexity; and the process underlying its growth has been driven in a chaotic fashion by the individual actions of numerous participants. Our experience with hits and pagerank suggests, however, that in many respects the end product is not as chaotic as one might suppose: the aggregate behavior of user populations on the www can be studied through a mathematically clean technique for analyzing the Web's link topology, and one can use this technique to identify themes about hyperlinked communities that appear to span a wide range of interests and disciplines.

REFERENCES

[1] S. Brin, L. Page, The anatomy of a large-scale hypertextual Web search engine, Computer Networks, 30(1 7): 107-117, 1998, Proceedings of the 7th International World Wide Web Conference(WWW7).
[2] J. Kleinberg. Authoritative sources in a hyperlinked environment,1997. Research Report RJ 10076 (91892), IBM.
[3] D. Gibson, J. Kleinberg, and P. Raghavan. Inferring Web communities from link topology. In Proc. 9th ACM Conference on Hypertext and Hypermedia (HyperText 98), pages 225–234, Pittsburgh PA, June 1998.
[4] http://www.altavista.com
[5] http://dustinbrewer.com/top-10-search-engine-optimization-techniques/
[6] SaekoNomura,SatoshiOyama,Tetsuo Hayamizu, Toru Ishida. Analysis and Improvement of HITS Algorithm for Detecting Web Communities. Department of Social Informatics, Kyoto University Honmachi Yoshida Sakyo-ku, Kyoto, 606-8501 Japan.

[7] AmirPanah, AminPanah. Evaluating the datamining techniques and their roles in Increasing the search speed data in web. 978-1-4244-5540-9/10/$26.00 ©2010 IEEE.
[8] G. Pinski, F. Narin, Citation influence for journal aggregates of scientific publications: Theory, with application to the literature of physics, in Information Processing and Management. 12, (1976).
[9] D7. Gibson, J. Kleinberg, P. Raghavan, Inferring Web Communities from Link Topology, in the Proceedings of the 9th ACM Conference on Hypertext and Hypermedia, (1998).
[10] M. Rajman, M. Vesely, From Text to Knowledge: Document Processing and Visualization: a Text Mining Approach, in Proceedings of the NEMIS Launch Conference, International Workshop on Text Mining & its Applications, Patras, Greece, April(2003).
[11] G. Salton and M. McGill. Introduction to Modern Information Retrieval. McGraw-Hill, 1983.