**Available Online at www.ijcsmc.com**

**International Journal of Computer Science and Mobile Computing**

**A Monthly Journal of Computer Science and Information Technology**

**ISSN 2320–088X**
**IMPACT FACTOR: 6.199**

***IJCSMC, Vol. 9, Issue. 2, February 2020, pg.137 – 146***

# HIDDEN WEB CRAWLER

**Ashwini Bhardwaj** (sonuashwini5@gmail.com)
Guide- **Dr. Shavita Shiwani; Vikas Verma**
Director – **Dr. Harvir Singh**

***ABSTRACT: Traditional search engines deal with the Surface Web which is a set of Web pages directly accessible through hyperlinks and ignores a large part of the Web called hidden Web which is a great amount of valuable information of online database which is "hidden" behind the query forms. To access to those information the crawler have to fill the forms with a valid data, for this reason we propose a new approach which use DIA and PSO technique in order to find the most promising keywords of a specific domain for automatic form submission. The effectiveness of proposed framework has been evaluated through experiments using real web sites and encouraging preliminary results were obtained.***

## I. INTRODUCTION:

**What is web search?**

- Access to "heterogeneous", distributed information
    1. Heterogeneous in creation
    2. Heterogeneous in motives
    3. Heterogeneous in accuracy …
- Multi-billion dollar business
- Source of new opportunities in marketing
- Strains the boundaries of trademark and intellectual property laws
- A source of unending technical challenges

**Brief (non-technical) history:**

- Early keyword-based engines
    1. Altavista, Excite, Infoseek, Inktomi, Lycos, ca. 1995-1997

- Paid placement ranking: Goto.com (morphed into Overture.com → Yahoo!)

2. Your search ranking depended on how much you paid

Auction for keywords: casino was expensive!

**Brief (non-technical) history**

- 1998+: Link-based ranking pioneered by Google

1. Blew away all early engines save Inktomi
2. Great user experience in search of a business model
3. Meanwhile Go to/Overture's annual revenues were nearing $1 billion

- Result: Google added paid-placement "ads" to the side, independent of search results

1. 2003: Yahoo follows suit, acquiring Overture (for paid placement) and Inktomi (for search)

**Architecture of Ranking**

Webpages that are highly ranked on many search engines are likely to be more relevant in providing useful information. However, all search engines have different ranking scores for each website and most of the time these scores are not the same. This is because search engines prioritise different criteria and methods for scoring, hence a website might appear highly ranked on one search engine and lowly ranked on another. This is a problem because web crawler engines rely heavily on the consistency of this data to generate reliable accounts.

**Web Search Ranking:**

A web search engine is a software system that is designed to search for information on the World Wide Web. The search results are generally presented in a line of results often referred to as search engine results pages (SERPs). The information may be a mix of web pages, images, and other types of files. Some search engines also mine data available in databases or open directories.

Search engine ranking refers to the position at which a particular site appears in the results of a search engine query. Each page of the search results typically lists 10 websites, although they are sometimes augmented with local listings, videos and images. A site is said to have a high ranking when it appears at or near the top of the list of results. Ranking higher in the search results actually corresponds to a lower number (#1), while ranking lower corresponds to a higher number (#10). Many site owners engage in SEO campaigns in order to improve their search engine ranking and move their website closer to the top of the results because websites that are ranked higher typically get a larger percentage of click-throughs and attract more visitors than lower ranked websites.

Search engine ranking is influenced by a multitude of factors including age of site, the quality of a site's link portfolio, relevancy of the page, social signals and level of competition, among others. Google admits to using 200 factors when determining a site's search engine ranking, many of which cannot be controlled by the website owner. Among other things, a white hat SEO campaign is designed to positively impact these factors and improve a website's overall search engine ranking.

The search engines are also focused on creating a more personalized search experience for users and take an individual's search history into account. While a website might rank #3 for certain search phrase for one visitor, it might have only ranked #8 for another. Search engine rankings are not guaranteed from one search to another, even when the search phrase is the same.

Search engines rank individual pages of a website, not the entire site. This means that the homepage might rank #1 for certain keywords, while a deep internal page might be listed on the third page.

**PAGE RANK:**

Page Rank is a numeric value that represents how important a page is on the web. Page Rank is the Google's method of measuring a page's "importance." When all other factors such as Title tag and keywords are taken into account, search engine uses Page Rank to adjust results so that more "important" pages move up in the results page of a user's search result display. Search Engine Figs that when a page links to another page, it is effectively casting a vote for the other page. It calculates a page's importance from the votes cast for it. How important each vote is taken into account when a page's Page Rank is calculated. It matters because it is one of the factors that determine a page's ranking in the search results. It isn't the only factor that Google uses to rank pages, but it is an important one. The order of ranking in Search Engines works like this:

1. Find all pages matching the keywords of the search.

2. Adjust the results by Page Rank scores.

Page Rank takes the back links into account and propagates the ranking through links. A page has a higher rank, if the sum of the ranks of its backlinks is high. The original Page Rank algorithm is given in following equation

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

Where,

PR (P) = Page Rank of page P

PR (Ti) = Page Rank of page Ti which link to page

C (Ti) = Number of outbound links on page T

D = Damping factor which can be set between 0 and 1.

**Page Ranking Algorithms**

Some of the prevalent page ranking algorithms have been discussed in this section. Some algorithms like PageRank rely on the link structure of the documents i.e. their popularity scores (web structure mining), whereas some utilize the content in the documents (web content mining), the usage browsing behavior (web usage mining), while some use a combination i.e. they use links as well as content or usage patterns of the document to assign a rank value to the concerned document.

**PageRank Algorithm:**

Google widely utilizes a ranking algorithm developed by Surgey Brin and Larry Page , named PageRank (PR) that uses the hyperlink structure of the web to assign an importance based sorting among the web pages. This algorithm states that if a page is getting some important incoming links

to it then its corresponding outgoing links also experience that level of importance. The algorithm helps propagate and distribute the rank among the pages via these links. Thus, a page would enjoy a higher rank if the sum of the ranks of its incoming links is high. After the user has fired the query, Google combines pre-computed static PageRank scores with content matching scores to obtain an overall ranking score for each web page

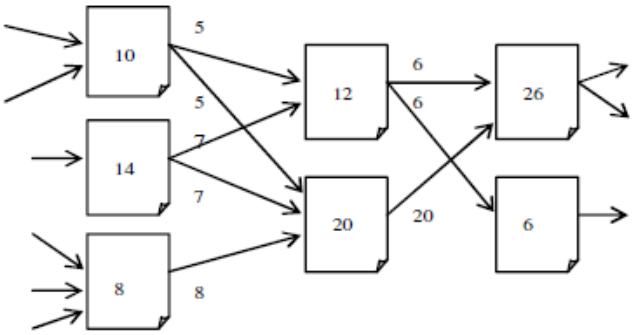



**Figure 3:** Equal distribution of Page| Ranks

A mathematical expression for the PageRank is defined in (1).

$$PR(u) = (1-d) + d \sum_{v \in B(u)} \frac{PR(v)}{N_v} \qquad (1)$$

here u is a web page, B(u) is the set of pages that representing back links for u, PR(u) and PR(v) are rank scores of page u and v respectively, Nv denotes the number of outgoing links of page v, d is a damping factor that can be thought of as the probability of users' following direct links i.e the web graph and is usually set to 0.85.

In PR, the rank score of a page (say p) is equally divided among its outgoing links and the values assigned to the outgoing links of page p are in turn used to calculate the ranks of pages pointed to by p, this equality in division became the shortcoming of PR.

**Weighted PageRank Algorithm:**

This algorithm is an extension of Page Rank algorithm. WPR takes into account the importance of both the in links and the out links of the pages and distributes rank scores based on the popularity of the pages. WPR performs better than the conventional Page Rank algorithm in terms of returning larger numbers of relevant pages to a given query. According to author the more popular web pages are the more linkages that other Web Pages tend to have to them or are linked to by them. The proposed extended Page Rank algorithm–Weighted Page Rank Algorithm assigns larger rank values to more important (popular) pages instead of dividing the rank value of a page evenly among its out link pages. Each out link page gets a value proportional to its popularity (its number of in links and out links). The popularity from the number of in links and out links is recorded as Win (v, u) and Out(v, u), respectively. WPR supplies the most important web pages or information in front of users.

Original Weighted PageRank formula is

$$PR(U)=(1-d)+d\sum_{v\in B(u)} PR(V)\,W^{in}(U,V)W^{out}(U,V) \quad \ldots(1)$$

Actually proposed by Wenpu Xing and Ali Ghorbani , the Weighted PageRank Algorithm (WPR) is an extension to standard PageRank. More specifically it is a modification in the rank distribution part of PR. It assumes that a more popular page must have more linkages than the other web pages tend to have to it or are linked to by it. This algorithm, instead of dividing the rank evenly among its outgoing linked pages, assigns larger rank values to more important pages. Each outgoing link gets a value proportional to its popularity or importance. The popularity of a page is measured by its number of incoming and outgoing links. The popularity is assigned in terms of weight values to the incoming and outgoing links and are denoted as Win(v, u) and Wout(v, u) respectively.

Win(v, u) (given in (2)) is the weight of link(v, u) calculated based on the number of incoming links of page u and the number of incoming links of all reference (outgoing linked) pages of page v.

$$W^{in}_{(v,u)} = \frac{I_u}{\sum_{p\in R(v)} I_p} \quad (2)$$

where Iu and Ip represent the number of incoming links of page u and page p, respectively. R(v) denotes the reference page list of page v. Wout(v,u) (given in (3)) is the weight of link(v, u) calculated based on the number of outgoing links of page u and the number of outgoing links of all reference pages of page v.

$$W^{out}_{(v,u)} = \frac{O_u}{\sum_{p\in R(v)} O_p} \quad (3)$$

where Ou and Op represent the number of outgoing links of page u and page p, respectively. Considering the importance of pages, the original PageRank formula (1) is modified as given in (4).

$$PR(u) = (1-d) + d\sum_{v\in B(u)} PR(v)W^{in}_{(v,u)}W^{out}_{(v,u)} \quad (4)$$

**Page Ranking based on Link-Visits (PRLV)**

PRLV (Page Ranking based on Link Visits) [10] is an efficient page ranking mechanism that roams around two major web mining techniques namely, the Web Structure Mining and Web Usage Mining. It takes the user visits of pages/links into account that directs the calculation of the grandness and relevance score of the web pages. The system constitutes many subtasks, till final rank gets determined, which are outlined below:

1. Storage of user's access information (hits) on an outgoing link of a page in related server log files.
2. Fetching of pages and their access information by the targeted web crawler.
3. For each page link, computation of weights based on the probabilities of their being visited by the users.
4. Final rank computation of pages based on the weights of their incoming links.
5. Retrieval of ranked pages corresponding to user queries.

To sum up, a ranking is assigned to pages according to clicks a page has got from users with respect to some query, and this rank, instead of being distributed evenly among pages, is assigned according to the weight their incoming links possess.

The weights are determined for out-links of pages (inner-document structure mining) and ranks are computed by taking back-links into account (inter-document structure mining).

Demonstrated the system into two phases wherein the first phase performs the first task, while the rest of four tasks are dedicated to the second phase.

The two phases as dictated in the work are:

1. Link-Visits (hits) Calculation
2. Rank Calculation and Result retrieval

**Link-Visits (hits) Calculation**

A user fires a web page request using its client which in turn asks a server side agent to retrieve the same from its store. The server along with query processing also requests the user's browsing information from the client with the help of which the user is searching. This phase results with the calculated number of visits (from different users) on outgoing links of web pages.

**Rank Score Calculation in PRLV:**

Crawlers are the web agents that regularly visit the web servers to grab the web pages from their store. Here in the system under discussion, the crawling functionality has been modified slightly so as to get the hits information along with the web pages. This information is fed to a new module called the PRLV calculator that outputs us with the ranks of the various pages. PRLV calculates these ranks on the pre mentioned technique i.e. taking the back links into account.

The ranks are incrementally propagated forward according to the web graph, assigning the weights to the links according to their probability being visited by the users. Thus, the weights from the back links are distributed to the forward links according to their importance rather than equal distribution as it was done by the earlier algorithms. The probabilistic distribution of ranks among pages counts with a generic ranking, making the more relevant and important pages to flow higher in the list.

Each link in the crawled web graph is assigned a weight, which indicates its probability of being visited by the users. Obviously, a link with high probability is considered more important than others. In the proposed system, a page assumes a high rank if its back-linked pages possess high weights. Thus, ranking is propagated iteratively through back links. The weight assigned to a link (p, o) is given by (5)

$$Weight_{link}(p,o) = \frac{VC(p,o)}{\sum_{o' \in O(p)} VC(p,o')} \quad (5)$$

where p and o are two pages, p is associated with the out-linked page set O(p), each out link is associated with a numerical integer indicating visit-count (VC).

It may be noted that each out link has a weight associated with it, which is different from the weights of other out links of the same page.

The page rank value of a page is calculated based on visits of inbound links. If p is a page having inbound-linked pages in set B(p), then the rank (PRLV) is given by:

$$PRLV(p) = (1-d) + d\left(\sum_{b \in B(p)} PRLV(b).Weight_{link}(b,p)\right) \quad (6)$$

where d is the damping factor as is used in PageRank, Weight link() is the weight of the link calculated by (5)

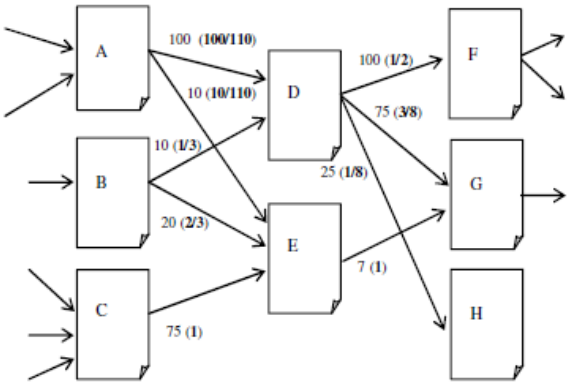



Figure: Hyperlink structure with link visit

It may be noted here that sum of weights of outbound links of a page is always 1. So, this mechanism provides an unequal distribution to importance of links as compared to PageRank method.

## II. RELATED WORK:

**Cheng Sheng [01],** A hidden database refers to a dataset that an organization makes accessible on the web by allowing users to issue queries through a search interface. In other words, data acquisition from such a source is not by following static hyper-links. Instead, data are obtained by querying the interface, and reading the result page dynamically generated. This, with other facts such as the interface may answer a query only partially, has prevented hidden databases from being crawled effectively by existing search engines. This paper remedies the problem by giving algorithms to extract all the tuples from a hidden database. Our algorithms are provably efficient, namely, they accomplish the task by performing only a small number of queries, even in the worst case. We also establish theoretical results indicating that these algorithms are asymptotically optimal – i.e., it is

impossible to improve their efficiency by more than a constant factor. The derivation of our upper and lower bound results reveals significant insight into the characteristics of the underlying problem. Extensive experiments confirm the proposed techniques work very well on all the real datasets examined.

**Manuel Álvarez [02]**, There is a great amount of valuable information on the web that cannot be accessed by conventional crawler engines. This portion of the web is usually known as the Deep Web or the Hidden Web. Most probably, the information of highest value contained in the deep web, is that behind web forms. In this paper, we describe a prototype hidden-web crawler able to access such content. Our approach is based on providing the crawler with a set of domain definitions, each one describing a specific data-collecting task. The crawler uses these descriptions to identify relevant query forms and to learn to execute queries on them. We have tested our techniques for several real world tasks, obtaining a high degree of effectiveness.

**Sweety Mangla [03]**, The traditional search engine work to crawl, index and query the "Surface web". The Hidden Web is the willing on the Web that is accessible through a html form. not general search engines. This Hidden web forms 96% of the total web .The hidden web carry the high quality data and has a wide coverage. So hidden web has always stand like a golden egg in the eyes of the researcher. This high quality information can be restored by hidden web crawler using a Web query front-end to the database with standard HTML form attribute. The documents restored by a hidden web crawler are more proper, as these documents are attainable only through dynamically generated pages, delivered in response to a query. A huge amount of data is hidden in the databases behind the search interfaces which need to be indexed so in order to serve user's query. To index these documents adequate, the search engine requires new indexing technique that optimizes speed and performance for finding relevant documents for a search query. Efficient index structures need to build to maintain the Hidden web data, which are then accessed to provide proper answers to many types of user's query.

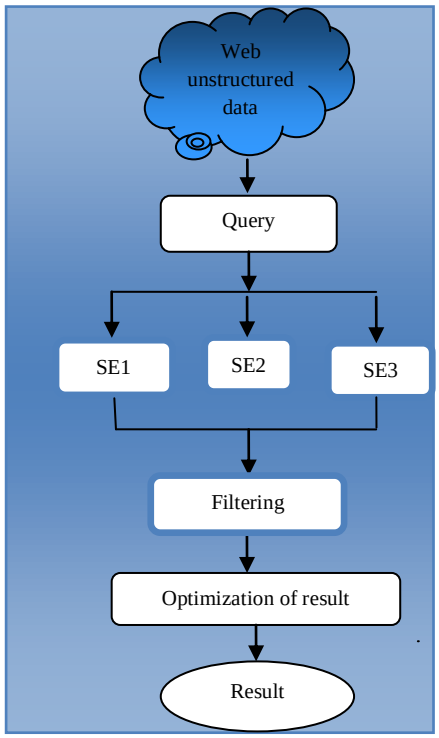
**Nimisha Jain [04]**, in a growing world of internet, thousands of web pages are added daily. But only approx. 0.03 percent fraction of web pages are retrieved by all the search engines. The remaining pages are deep websites. Deep web is that part of web which is hidden and unrecognizable by the existing search engines. It has been a longstanding challenge for the existing useful crawlers and web search engines to harvest this ample volume of data. This paper surveys on different methods of crawling deep-web. Density and rapidly changing nature of deep-web has posed a big hurdle in front of researchers. To overcome this issue, we propose a dual-layer framework, namely Smart Web Crawler, for efficiently harvesting deep web interfaces. Smart crawler consists of two layers: Site-discovery and In-depth Crawling. Site-discovery layer finds the sparsely located deep websites from given known parent sites using Reverse Searching and focused crawling. The In-depth Crawling layer makes use of Smart Learning and Prioritizing to crawl hyperlinks within a site to ensure wider coverage of web directories.

**Sudhakar Ranjan [05]**, Web is title admittance today mainly relies on search engines. A large amount of data is hidden in the databases behind the search interfaces referred to as "Hidden web", which needs to be indexed so in order to serve user's query. In this paper database and data mining techniques are used for query interface integration (QII). The query interface must resemble the look and feel of local interface as much as possible despite being automatically generated without human support. This technique keeps the related documents in the same domain so that searching of documents becomes more efficient in terms of time complexity.

## III. PROPOSED METHODOLOGY:

In our proposed system we take result from different existing search engines and store in database. By extracting different results from different search engine for one query the output is not same for all search engines it may be differ at rank wise. When we store result in database but here is redundancy in stored result to eliminate this redundancy we proposed DIA (Duplicacy Identification Algorithm) Algorithm. After that main purpose of our research work is optimize the result so that we use particle swarm optimization technique and compare that result to exiting technique. Both DIA and PSO gives the better accuracy and effective result from previous method.




## IV. CONCLUSION:

In this Paper, ranking of recovered URLs for different search engine is talked about. The work includes, by analyzing retrieved URL from multiple search engines based technique for positioning recovered results from various web search tools for viable presentation of query items to expand reliability of multiple search engine results.

# REFERENCES

**[1].** Cheng Sheng, Nan Zhang, Yufei Tao, Xin Jin, **"Optimal Algorithms for Crawling a Hidden Database in the Web",** National Research Foundation of Korea, and funded by the Ministry of Education, Science and Technology of Korea (Project No: R31-30007).

**[2].** Manuel Álvarez, Juan Raposo, Fidel Cacheda and Alberto Pan, **"A Task-specific Approach for Crawling the Deep Web"**, Engineering Letters, 13:2, EL_13_2_19 (Advance online publication: 4 August 2006).

**[3].** Sweety Mangla, Geetanjali Gandhi, **"Study of Crawlers and Indexing Techniques in Hidden Web"**, International Journal of Computer Science and Mobile Computing, Vol.4 Issue.4, April- 2015, pg. 598-606.

**[4].** Nimisha Jain, Pragya Sharma, Saloni Poddar, Shikha Rani, **"Smart Web Crawler to Harvest the Invisible Web World"**. International Journal of Innovative Research in Computer and Communication Engineering, Vol. 4, Issue 4, April 2016, ISSN (Print) : 2320-9798.

**[5].** Sudhakar Ranjan, Komal K. Bhatia, **"QUERY INTERFACE INTEGRATOR FOR DOMAIN SPECIFIC HIDDEN WEB",** International Journal of Computer Engineering & Applications, Vol. IV, Issue I/III, ISSN : 2321-3469.

**[6].** Eric J. Glover, Steve Lawrence, William P. Birmingham, C. Lee Giles, **"Architecture of a Metasearch Engine that Supports User Information Needs",** CIKM '99 11/99 Kansas City, MO, USA, 1999 ACM l-581 13-075-9/99/0010.

**[7].** Choon Hoong Ding and Rajkumar Buyya, **"Guided Google: A Meta Search Engine and its Implementation using the Google Distributed Web Services".**

**[8].** B. Uygar Oztekin George Karypis Vipin Kumar, **"Expert Agreement and Content Based Reranking in a Meta Search Environment using Mearf",** WWW2002, May 7–11, 2002, Honolulu, Hawaii, USA, ACM 1-58113-449-5/02/0005.

**[9].** Lin Deng, Xiaoyong Chai, Qingzhao Tan, Wilfred Ng, Dik Lun Lee, **"Spying Out Real User Preferences for Metasearch Engine Personalization".**

**[10].** Hossein Jadidoleslamy, **"Search Result Merging and Ranking Strategies in Meta-Search Engines: A Survey",** International Journal of Computer Science Issues, Vol. 9, Issue 4, No 3, July 2012, ISSN: 1694-0814.

**[11].** Adrian P. O'Riordan, **"Open Meta-search with OpenSearch: A Case Study",** Computer Science Department, University College Cork, Cork, Ireland. December 2007.

**[12].** Zonghuan Wu, Weiyi Meng, Clement Yu, Zhuogang Li, **"Towards a Highly-Scalable and Effective Metasearch Engine",** WWW10, May 1-5, 2001, Hong Kong, ACM 1-58113-348-0/01/0005.

**[13].** K.Srinivas, P.V.S. Srinivas, A.Govardhan, **"Web Service Architecture for a Meta Search Engine",** (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 2, No. 10, 2011.

**[14].** Mandis Beigi, Ana B. Benitez, and Shih-Fu Chang, **"MetaSEEk: A Content-Based Meta-Search Engine for Images".**