Hidden Web Data Extractor
Komal Taneja
MDU Rohtak
Shri Ram College of Engineering, Palwal
gandhi1211@gmail.com

Puneet Rani
MDU Rohtak
Shri Ram College of Engineering, Palwal
puneetrani.5270@gmail.com

Abstract
In present scenario, WWW is very important and most common place for the users of the internet. We are facing a world with ever growing data. This data is stored in documents in networks. The most prominent of these networks is the World Wide Web often referred to as the Internet. Finding and retrieving information or documents in a large environment is a huge task. For this reason crawlers have been developed. They traverse over a subset of the Internet, and try to index the pages according to their relevance. Due to the fact that the Internet is huge and ever growing, crawlers can only view a subset of the internet. Thus one large problem of crawlers is to select the relevant pages of the Internet and index them accordingly. The modern Internet nowadays also contains multimedia contents like video images and audio files. Extracting information from these data sources and indexing these is a major challenge for modern crawlers. The aim of our research is to develop a crawler capable of querying and extracting the data from the hidden web database. This system will allow a user to submit desired query or keywords on a form-based interface and crawler extract required data from the hidden web not the hyperlinks that traditional crawlers does.

Keywords Used-
Surface web, Hidden web Crawler, web Content, WWW

Introduction
Crawlers are programs that automatically traverse the Web, retrieving pages and building a local repository of the portion of the web that they visit. It indexes all the words in a document, adds them to a database, then follows all hyperlinks and indexes and adds that information also to the database. A Web crawler systematically browses the World Wide Web, typically for the purpose of Web indexing. A Web crawler may also be called a Web spider, an ant, an automatic indexer or a Web scatter. Web search engines and some other sites use Web crawling or spidering software to update their web content or indexes of others sites' web content. Web crawlers can copy all the pages they visit for later processing by a search engine that indexes the downloaded pages so that users can search them much more quickly.

Current-day crawlers retrieve content only from the publicly index able web, i.e., the set of web pages reachable purely by following hypertext links, ignoring search forms and pages that require authorization or prior registration. In particular, they ignore the tremendous amount of high quality content “hidden” behind search forms. The hidden Web continues to grow, as organizations with large amounts of high-quality information (e.g., the Census Bureau, Patents and Trademarks Office, news media companies) are placing their content online, providing Web-accessible search facilities over existing databases. Under this, I will go to build a hidden Web crawler; that can crawl and extract content from these hidden databases. Such a crawler will enable indexing, analysis, and mining of hidden web content. In particular, a large part of the Web is “hidden” behind search forms and is reachable only when users type in a set of keywords, or queries, to the forms. These pages are often referred to as the Hidden Web [11] or the Deep Web [7], because search engines typically cannot index the pages and do not return them in their results.
There are many general purpose crawlers, some of them are listed below- [1]:

- **Yahoo Slurp** is the Yahoo’s! Search crawler.
- **Googlebot** is the Google’s crawler.
- **Bingbot** is the Microsoft’s bing web crawler.
- **RBSE** was the first published web crawler.
- **WebFountain** is a distributed modular crawler.
- **Scooter** is the AltaVista Web crawler.

A crawler is a program that downloads and stores Web pages, often for a Web search engine. Roughly, a crawler starts off by placing an initial set of URLs in a queue, where all URLs to be retrieved are kept and prioritized. From this queue, the crawler gets a URL (in some order), downloads the page, extracts any URLs in the downloaded page, and puts the new URLs in the queue. This process is repeated. Collected pages are later used for other applications, such as a Web search engine or a Web cache. With web crawling, the general processes that a crawler takes are as follows:

- Check for the next page to download - the system keeps track of pages to download in a “queue”
- Check to see if the page is "allowed" to be downloaded - checking a “robots exclusion” file and also reading the header of the page to see if any exclusion instructions were provided do this. Some people don't want their pages archived by search engines.
- Download the whole page.
- Extract all links from the page (additional web site and page addresses) and add those to the queue mentioned above to be downloaded later.
- Extract all words, save them to a database associated with this page, and save the order of the words so that people can search for phrases, not just keywords
- Optionally filter for things like adult content, language type for the page, etc.
- Save the summary of the page and update the "last processed" date for the page so that the system knows when it should re-check the page at a later date.

**Literature Review**

Crawling deep web [2, 5, 6, 7, 11] is the process of collecting hidden data by issuing queries through various search interfaces including HTML forms, web services and programmable web APIs. Crawling deep web data sources is important for several reasons, such as indexing deep web data sources, or backing up data.

Recent studies show that a significant fraction of Web content cannot be reached by following links [4, 5]. According to many studies, the size of the Hidden Web increases rapidly as more organizations put their valuable content online through an easy-to-use Web interface [4]. In [5], Chang et al. estimate that well over 100,000 Hidden-Web sites currently exist on the Web. Moreover, the content provided by many Hidden-Web sites is often of very high quality and can be extremely valuable to many users [4]. For example, Pub Med Deep web is also known as ‘invisible web’ as these pages are out of reach from traditional search engines. It is also called as hidden web, deep net or Undernet [1]. In 1994, Dr. Jill Ellsworth “first coined the phrase “invisible Web” to refer to information content that was “invisible” to conventional search engines [3]. In 2001, Bergman coined it as “Deep Web”.

Raghavan and H. Garcia-Molina proposed architecture of the deep web crawler in the paper Crawling the hidden web (2001) [9]. In this paper, they gave a prototype for a hidden web crawler named HiWE. This is based on the concept of extraction of task specific information. The main advantage of this strategy is that it minimizes the extraction of relevant information. The limitations of HiWE include its inability to support partially filled forms and to identify dependence between certain elements of form. L. Barbosa and J. Freire proposed a crawling strategy to crawl the deep i.e. hidden web in the paper Searching for hidden web databases (2005) [13]. In this paper, they proposed a focused crawler based on the concept of automatic form filling. This technique has the main advantage of saving resources and time. R. Anita et al. proposed an intelligent deep web crawler Deep iCrawler, in the paper Deep iCrawler: An Intelligent Vision Based Deep Web Crawler (2011) [8], that is purely vision based. In this, they have tried to use the visual similarities of the data elements. There are three phases in Deep iCrawl –
Dilip Sharma and A.K. Sharma proposed a architecture for deep web crawler in their paper A Novel Architecture for Deep Web Crawler (2011) [20]. To minimize limitations of existing deep Web crawlers, a novel architecture was proposed based on QIIIEP specification. The proposed architecture is cost effective and has features of privatized search and general search for deep Web data hidden behind html forms. [20] Namrata and Dr. Subhash also proposed the architecture of a deep web crawler, in their paper Deep Web Crawl for Deep Web Extraction (2013) [14] for extracting information from deep web, which is based on the concept of multiple HTTP connections to WWW. This system is designed to be used on the client side [14].

Deep web v/s surface web
Below is the distinction between surface web and deep web-

**Surface web**
The surface Web (also known as the Clearnet, the visible Web or indexable Web) is that portion of the World Wide Web that is indexable by conventional search engines. The part of the Web that is not reachable this way is called the Deep Web. Search engines construct a database of the Web by using programs called spiders or Web crawlers that begin with a list of known Web pages. The spider gets a copy of each page and indexes it, storing useful information that will let the page be quickly retrieved again later. Any hyperlinks to new pages are added to the list of pages to be crawled. Eventually all reachable pages are indexed, unless the spider runs out of time or disk space. The collection of reachable pages defines the Surface Web.

**Deep web**
The Deep Web (also called the Deepnet, Invisible Web, or Deep Web) is World Wide Web content that is not part of the Surface Web, which is indexed by standard search engines. It should not be confused with the dark Internet, the computers that can no longer be reached via the Internet, or with a Darknet distributed filesharing network, which could be classified as a smaller part of the Deep Web.

Web searchers don’t get much relevant information as traditional search engines cannot find such specific and depth content which is demanded by the user. Traditional Search engines only searches static pages and those pages which are linked to one another .Even with the help of web crawler ,they cannot find dynamic pages which are generated dynamically or generated only by user specific query and closed or get invisible after the user query is stopped. But with the help of deep web crawling technique, most search engines can now search in depth or even those dynamic pages which were invisible while searching by using web crawling techniques.

Deep web is also known as ‘invisible web’ as these pages are out of reach from traditional search engines. It is also called as hidden web, deepnet or undernet . In 1994, Dr. Jill Ellsworth “first coined the phrase “invisible Web” to refer to information content that was “invisible” to conventional search engines. In 2001, Bergman coined it as “Deep Web”.

In brief, deep web, part of Internet, can be defined as the information concealed behind Hyper Text Markup Language (HTML). It is also believed that deep web is a vast source of methodized content on the World Wide Web and to access contents of deep web has been a challenge in the web community. Surface web is also called as open web, visible web or indexable net which can be accessed by the traditional search engines. Search engines access those web pages and websites which are linked to one another and are generally static and accessible to all web users. Approximately 80% of the information on the Web belongs to the ’invisible web’. The concept used behind surface web is to perform indexing of web pages and store them for future processing also.

**Limitation of Existing Web Crawler**
Due to the size of the Deep Web, it is not easy to develop a crawler that can automatically extract the contents from the hidden or deep web. For this reason, my efforts on developing a domain-specific querying system. The large amount of information on web is stored in backend databases which are not indexed by traditional search engines. Such databases are referred to as Hidden web databases and extraction of this hidden web content is a potential research area as the pages are dynamically created through search query interfaces. However, direct query through this search interface is laborious way
to search. Hence, there has been increased interest in retrieval and integration of hidden web data with a view to give high quality information to the web user. This paper proposes a novel approach that identifies Web page templates and the tag structures of a document in order to extract structured data from hidden web sources as the results returned in response to a user query are typically presented using template generated Web pages. Hidden web data, stored in structured or unstructured databases, is inherently hidden behind search forms. It is qualitatively and quantitatively different from the surface Web. The quality content of the deep Web is 1,000 to 2,000 times greater than that of the surface Web whereas overall the hidden web contains approximately 7,500 terabytes of data and 550 billion individual documents in contrast to the surface Web, which is reported to about 167 terabytes. Since the hidden web is the biggest source for structured data and is not publicly indexed yet, accessing the same is a challenging task especially when the pages are created dynamically through search interfaces. The traditional search engines use inverted index as a data structure to index the web data and keyword interface to retrieve the data. But, surfacing the Hidden Web is a more difficult task in many respects. First, the index structures for the hidden web deal with the structured data as well as the large volume of data. Second, the search query interfaces often have more than one attribute and requires their respective values to be submitted.

**Architecture for data extraction**

The goal of this paper is to focus on the working method of deep web crawler to extract the data from various hidden web databases and this data in integrated form will be stored in large repository with no duplicate records.

Since the hidden web is the biggest source for structured data and is not publicly indexed yet, accessing the same is a challenging task especially when the pages are created dynamically through search interfaces. The problem is how to design an interface for hidden web that can take query respective to any domain and return the result corresponding to that domain only. This Project proposes the solution to it. First task is to read the files containing the data of hidden web that is in structured form of various different domains and to index this data. Then a search query interface is designed that can take the query regarding any domain and return the data specific to that domain only. Several online databases provide dynamic query-based data access through their query interfaces, instead of static URL links. This Query interface is considered as an entrance to Hidden Web, as the tremendous amount of information is hidden behind these search forms in web pages and traditional crawler cannot replicate the query submission carried out by human beings. A Web search interface for e-commerce typically contains some HTML form control elements such as textbox i.e. a single line text input, radio button, checkbox and selection list i.e. a pull-down menu that allow a user to enter search information. A Search interface would provide uniform access to the data sources of a given domain of interest. Here, a Query is considered which contains some terms and it should not be blank. The User enters the query in the text box of the Search Query interface which is then accessed by Query Processor. The two main steps done by Query Processor are extracting the Query String and then tokenizing it. The Tokens are then matched with the attribute values stored in the repository to retrieve the posting lists. These lists are then intersected by the query processor to return the result to the user with the help of a Result Page. This architecture is shown in below figure 1.1.

There are mainly five components of the Proposed Architecture of Data Extraction as shown below:

**QUERY INTERFACE:**- A Query Interface is designed as an entrance to the Project which is used by the user to enter the query. It is the user interface where the user has to input the query to get the required data. This interface contains a Text Box where the user has to place the query.

**RESULT PAGE:**- After the query has been processed by the query processor and the Attribute-value based Searcher, the required data is returned to the user in the form of result page.

**QUERY PROCESSOR:**- The user queries are processed by the Query Processor to fetch the desired data and return it back to the user with desired results. It has the following two components:

**EXTRACT QUERY STRING:**- The Extract Query String module of the Query Processor extracts the
Query String from the Query Interface and passes it to the Attribute-value based Tokenization module.

ATTRIBUTE-VALUEBASED TOKENIZATION:- It tokenizes the Query string into a number of tokens. It passes the various tokens to the Attribute-value based Searcher for searching.

Attribute-value based Searcher matches the tokens respective to the domain with the various attribute and their corresponding values stored in the Data Repository to retrieve the postings lists which are then intersected to return the result page to the user.

DEEP WEB DATABASE:- The Data Repository stores the various postings lists of the attribute values along with the attributes and their respective domain. The Search interface allows a user to search some set of items without altering them. The user enters a query by typing to get the data of interest. Result Page is a page containing data of interest. This Project implements attribute-value based domain-specific searching technique for hidden web.

OFFLINE SEARCH:- In this project user can search either online or offline. For online searching process is described above. In offline searching, user can enter the query in textboxes and then search in the local database for the result. During online searching when result page is displayed then that result is also saved in the local database that is already designed for saving the retrieved data. Database is designed first according to the retrieved attributes. Data extracted and inserted under the respective column names in the table and values under each column will be stored as row.

CONCLUSION
The paper makes an analytical survey of several proven web crawlers capable of searching hidden pages. It also addresses the prospects and constraints of the methods and the ways to further enhance. To minimize user effort, the problem of automatically interaction with hidden web sources is explored. In this paper hidden web data extraction method has been discussed. Deep web has plentiful information contained in it. It is a repository of very useful contents that are important for researchers at many levels. To use these resources, there is need of an efficient method to get the relevant and desired content which is lying beneath the surface web i.e. deep web. Although some very useful algorithms and software have also been designed to explore the hidden web, yet there is much scope of finding new methods of crawling the deep web that can be cost and time effective.

REFERENCES
[6] Stephen W. Liddle, David W. Embley, Del T. Scott, Sai Ho Yau, Extracting Data behind Web Dynamic Query Interface...


