Deep Questions in the "Deep or Hidden" Web

Sonali Gupta and Komal Kumar Bhatia

Abstract The Hidden Web is a part of the Web that consists mainly of the information inside databases, i.e., anything behind an interactive electronic form (search interfaces), which cannot be accessed by the conventional Web crawlers [1, 2, 8]. However, there have been well-defined, effective, and efficient methods for accessing Deep Web contents. One of these methods for accessing the Hidden Web employs an approach similar to 'traditional' crawling but aims at extracting the data behind the search interfaces or forms residing in databases. The paper brings insight into the various steps, a crawler must perform to access the contents in the Hidden Web. We structure the problem area and analyze what aspects have already been covered by previous research and what needs to be done.

Keywords WWW · Hidden web · Surface web · Hidden web crawler

1 Introduction

The growth of the World Wide Web (WWW) has been phenomenal over the years [8, 10, 11]. Surface Web refers to the abundant web pages that are static, typically having, outgoing links to other web pages, and incoming links which allow them to be reached from other pages, creating a spider-web like system of interconnected data; whereas the Hidden Web (HW) consists of unlinked data and refers to the Web pages created dynamically as the result of a specific search. The Hidden or Deep Web consists mainly of information inside databases, i.e., anything behind an

S. Gupta (✉) · K. K. Bhatia
Department of Computer Engineering, YMCA University of Science and Technology, Faridabad, India
e-mail: Sonali.goyal@yahoo.com

K. K. Bhatia
e-mail: Komal_bhatia1@rediffmail.com

B. V. Babu et al. (eds.), *Proceedings of the Second International Conference on Soft Computing for Problem Solving (SocProS 2012), December 28–30, 2012*, Advances in Intelligent Systems and Computing 236, DOI: 10.1007/978-81-322-1602-5_87,

interactive electronic search form interface with most of it elicited by the HTTP form submission. Examples of Hidden web content include directories and collection of patents, scientific and research articles, holiday booking interfaces, etc. Estimates of the size of the Hidden Web differ, but some place it at up to 500 times the size of the traditional surface Web [1, 3, 5, 7].

The Hidden Web though hidden and not accessible through traditional document-based search engines, is a huge and distributed repository of data lying in databases which has to be accessed by some means. Methods must exist to prove the expediency of the source [8]. There are two basic approaches to access the contents in the HW:

1. Crawling/Trawling or Surfacing: It refers to the crawler's activity of collecting in the background as much relevant, interesting fraction of the data as possible and updating the search engine's index. This approach has the main advantage of best fit with the conventional search engine technology.
2. Virtual Data Integration: It refers to the creation of vertical search engines for specific domains where APIs will be used to access Hidden Web sources at time and construct the result pages based on their responses. Since external API calls need to be made by the search engine, this approach is traditionally slower than crawling.

The major goal of the paper is to describe the research problems associated with Hidden Web crawlers and analyze the existing research in the context of the research problems so as to identify and bring outstanding issues to the forefront.

2 Background (Search Engines/Crawlers)

Finding or Searching information on the Web has become an important part of our daily lives and about 550 million Web searches are performed every day [10, 11]. The tools that have been used to find information on the Web are typically known as Web Search Engines [2, 10, 11]. Figure 1 illustrates the activities and the corresponding components or elements of a basic search engine.

The various activities performed by a search engine can be divided into: *Crawling* by which a search engine gathers pages from the WWW; *Indexing* which is building a data structure that will allow quick searching of the text [11]; or "the act of assigning index terms to documents" where an index term is a (document) word whose semantics helps in remembering the document's main themes [11]; *Query Processing which* includes receiving a query from the user, searching the index or database for relevant entries, and presenting the results to the user. The component responsible for the process of crawling is known as a Crawl Engine or more typically a Web crawler [2, 11], whereas the element responsible for building the search engine's index is termed as the Index Engine or typically as an Indexer.

The increasing prevalence of online databases has influenced the structure of the web crawlers and their capabilities for information access through search form

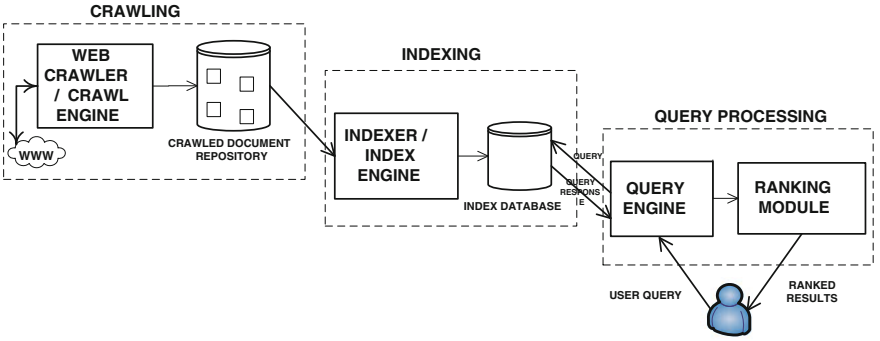

Fig. 1 Elements of a basic search engine

interfaces [2]. So, the paper discusses crawling, distinguishing on the basis of Surface and Hidden web crawl. The common belief is that over 1 million search engines currently operate on the WWW [7, 11] most of which cover only a small portion of the Web in their indices or databases. This coverage can be increased by either of the following means:

1. Employing multiple search engines: A search system that uses other search engines to perform the search and combines their search results is generally called a meta-search engine.
2. Enhancing the crawler's Capability: Since the HW comprises a major part of the Web (almost $\approx 80\,\%$) developing hidden Web crawlers has clearly become the next frontier for information access on the Web.

3 Hidden Web Crawler

Figure 2 illustrates the sequence of steps that take place when a user wants to access the contents in any Hidden Web resource. The user has to fill out a query form for retrieving documents that have been dynamically generated from the underlying database [3, 4].

Figure 3 illustrates the difference in the sequence of steps undertaken by any crawler to access the Hidden Web's informational content.

A Hidden Web crawler starts the same as the Surface web crawler by downloading the required web page, but then later it requires a lot of analysis and intelligence to extract information from the hidden web. The Surface Web Crawlers can record the address of a search front page but can tell nothing about the contents of the database [1, 5, 8].

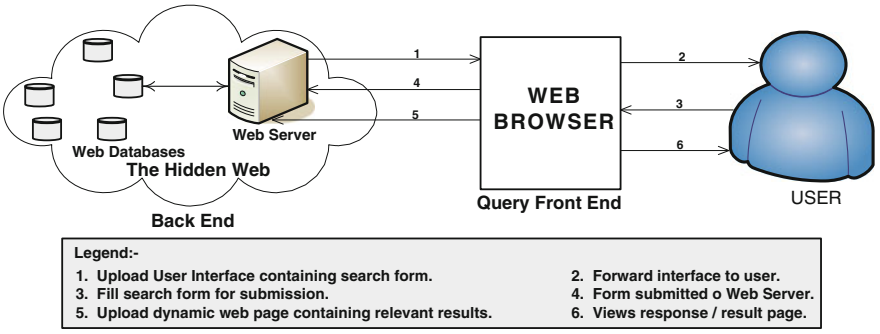

Fig. 2 User interaction with a search form interface

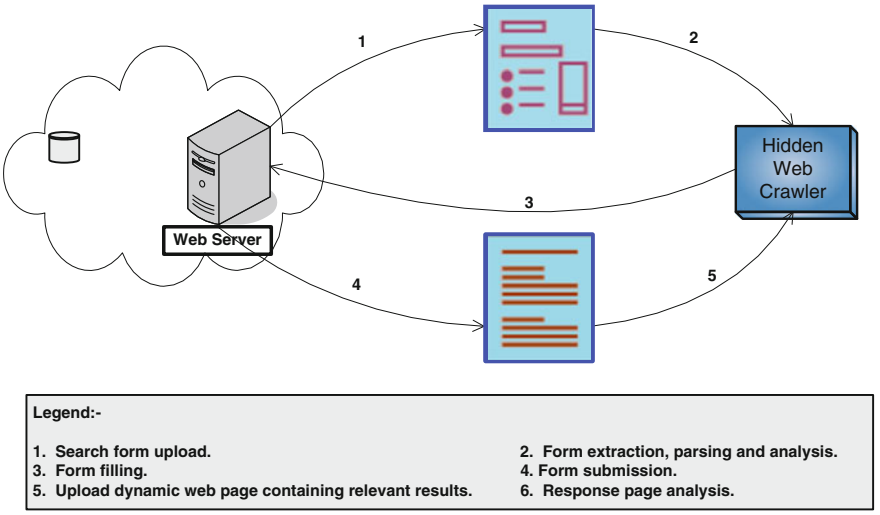

Fig. 3 A crawler interacting with the search form interface

4 Research Problems

Making a comprehensive crawl of the HW does not seem practical due to the two fundamental reasons [1, 3, 4, 8]: *Scale* The Hidden Web is unprecedented in many ways; unprecedented in size and content quality; unprecedented in the lack of coordination in its creation (distributed nature), and unprecedented in the diversity of backgrounds and motives of its participants. *Restricted Search form interfaces*: Access to the Hidden Web databases is provided only through restricted search interfaces, perceived to be used by humans [3, 4]. This raises the non-trivial problem of "training" the crawler for an appropriate use of the restricted interface to extract relevant content. Below are presented the two sub-problems or steps in the present scenario that suggest likely directions:

1. Resource discovery: In order to overcome the problem of Scale, the crawler must be trained to carry a crawl of only the relevant sources (effective crawling) rather than carrying a comprehensive crawl of the Hidden Web (exhaustive crawling) [3, 4, 8, 16, 17]. This requires the crawler to first locate the sites containing search form interfaces and then select the relevant subset from it. And as the Hidden Web data sources are growing continuously at a high rate, selecting the subset of relevant sources will prove not only cost-effective but also effective in time and make the crawler less prone to errors.
2. Content extraction: The task of harvesting information lying behind the form interfaces of the selected Hidden Web sources depends largely on the way, the crawler is able to understand and model the search form interface so as to come up with meaningful queries to issue to the search interface for probing the database behind it. The crawler must then be able to extract the data instances from the retrieved result pages. This problem of Content Extraction poses significant challenges and the solution lies in the three steps process comprising: Understanding the Search Form interfaces, automatically filling them and Information extraction.

The three steps together form the following basic modules of the system: *Form Analyzer* that will analyze each and every downloaded page to see if it can be used as a search page to retrieve information or not. It basically checks whether a web page is query able, has some form fields or not; *Form Parser* that extracts the fields from the search form and passes them on to the form processor for filling; *Form Processor* that fills in the various form fields by assigning appropriate values and finally submits the form for retrieving result pages; *Result Analyzer* that will analyze all the result pages obtained by the crawler after form execution, in order to get the required information.

5 The State of the Art in Hidden Web

In this section we discuss the previous work in the area grouped by the problem domains. Research on Hidden Web search can be dated back to the 1980s. Since then, substantial progress has been made in different sub-problems of crawling and accessing the Hidden Web.

5.1 Resource Discovery

The goal of any focused crawler is to select links that lead to documents that have been identified as relevant to the topic of interest and hence addresses the resource discovery problem, The work on focused crawling [14, 16, 17] describes the design of topic-specific crawlers for the Surface Web which complements our problem, as

same resource discovery techniques can be used to identify target sites for a Hidden Web crawler. The work in [14] discusses a best-first focused crawler which uses a page classifier to guide the search. Unlike an exhaustive crawler which follows each link in a page in a breadth first manner, this crawler gives priority to links that belong to pages classified as relevant. In domains that are not very narrow the number of irrelevant links can still be very high, the strategy can lead to suboptimal harvest rates and an improvement to which was proposed in [17].

An issue that remains with these focused crawlers is that they may miss relevant pages by only crawling pages that are expected to give immediate benefit. In order to address this limitation, certain strategies have been proposed that train a learner by collecting features from paths leading to a page, as opposed to just considering the contents of a page [17–19]. Reference [19] extends the idea in [14] and presents a focused crawling algorithm that builds a model for the context within which topically relevant pages occur on the Web. Another extension of the focused crawler idea is presented in [18] using a reinforcement learning algorithm to develop an efficient crawler for building domain-specific search engines.

Finally, it is worth pointing out that there are directories specialized on hidden-Web sources, e.g., [1, 20] that organize pointers to online databases in a searchable topic hierarchy and hence can be used as seed points for the crawl.

5.2 Content Extraction

Extracting content from the Hidden Web has received a lot of attention to date [3–6, 12, 15]. Most approaches to information retrieval in the Hidden Web are focused on understanding the various semantics associated with the form elements and automatically filling them as they are the only entry points to the Hidden Web.

Reference [3] presents an architectural model for a task-specific semi-automatic Hidden-Web crawler. The main focus of their work is to learn Hidden-Web query interfaces, not to generate queries automatically. A significant contribution of this work is the label matching approach that identifies elements of a form based on layout position, not proximity within the underlying HTML code. When analyzing forms, HiWE only associates one text to each form field according to a set of heuristics that take into account the relative position of the candidate texts with respect to the field (texts at the left and at the top are privileged), and their font sizes and styles. To learn how to fill in a form, HiWE matches the text associated with each form field and the labels associated to the attributes defined in its LVS. In this process, HiWE has the following restriction: it requires the LVS table to contain an attribute definition matching with each unbounded form field.

Many approaches exist that rely on filling forms [4, 6, 15] automatically. The main focus of the work in [4] is to generate queries automatically without any human intervention in order to crawl all the content behind a form. New techniques are proposed to automatically generate new search keywords from previous results, and to prioritize them in order to retrieve the content behind the form, using the minimum

number of queries. The problem of extracting the full content behind a form has been also addressed in [6]. They have proposed a domain-independent approach for automatically retrieving the data behind a given Web form. The approach to gather data is based on two phases: first the responses from the web site of interest are sampled and then if necessary methodically try all possible queries until either a fix point of retrieved data has arrived or all possible queries have exhausted. They have developed a prototype tool that brings the user into the process when an automatic decision becomes hard to make. These techniques focus on coverage, i.e., retrieve as big a portion of the site's content as possible.

The hidden web can also be accessed using the meta-search paradigm instead of the crawling paradigm. This body of work is often referred to as meta-searching or database selection problem over the Hidden Web. In meta-search systems, a query from the user is automatically redirected to a set of underlying relevant sources, and the obtained results are integrated to return a unified response. Data integration is the problem of combining data from various web databases sources to provide the users with a unified view of data [8]. One of the main tasks to formalize the design of a data integration system is to establish the mapping between the Web database sources and a global schema. The meta-search approach is more lightweight than the crawling approach, since it does not require indexing the content from the sources. Nevertheless, the users will get higher response times since the sources are queried in real-time.

6 Open Issues in Hidden Web Crawling

A critical look at the available literature indicates that the following issues need to be addressed while designing the framework for any fully automatic crawler for the Hidden Web. Most of the research to date has focused on the last issue. Little attention has been made to the first two questions of scalability and synchronization:

1. There exists a variety of Hidden Web sources that provide information about the multitude of topics/domains [1, 7, 20]. The continuous growth of information about the WWW [8, 10] and hence the domain-specific information with ever-increasing number of domain areas pose a challenge to crawler's performance. The crawl of the portion of the web for a particular domain must be completed within the expected time. This download rate of the crawler is limited by the underlying resources. An open challenge is the design a crawler that scales its performance according to the increase in the information on the WWW and number of domains. These scalability limitations stem from search engines' attempt to crawl the whole Web, and to answer any query from any user.
2. Decentralizing the crawling process is clearly a more scalable approach and bears the additional benefit that crawlers can be driven by a rich context (topics, queries, user profiles) within which to interpret pages and select the links to be visited. However, a rigorous focus only on scalability can be costly; of course,

the system must also coordinate information coming from multiple sources, not all of which are under the control of the same organization. The pattern of communication is many-to-many, with each server talking to multiple clients and each client invoking program on multiple servers.

3. As the number of data sources is growing continuously at a very high rate, it is very tedious, time-consuming, and error-prone to process the search interfaces in web-based applications. An important objective of any Hidden Web crawler is to build an internal representation of these search forms [4–6] that supports efficient form processing and interface matching techniques so as to fully automate the process.

7 Conclusion and Future Work

A move in the Web structure from hyperlinked graph in the past to electronic form-based search interfaces of present day, represent the biggest challenge a Web crawler needs to tackle with. Despite the Web's great success as a technology and the significant amount of computing infrastructure on which it is built, it remains as an entity, surprisingly unstudied. Users need and want better access to the information on the Web. We believe that Hidden Web crawling is an increasingly important and fertile area to explore as such a crawler will enable indexing, analysis, and mining of Hidden Web content, akin to what is currently being achieved with the Surface Web. The paper provides a look at some of the technical challenges that must be overcome to model the Web as a whole, keep it growing and understand its continuing social impact. The topic of concerns as mentioned in the paper are further exacerbated by the rapid growth of Hidden Web content, fueled by the success of social networking online, the proliferation of Web 2.0 content and the profitability of the companies that steward in this new era. We look forward to continuing this promising line of research. One of the main objectives of our work will remain as the design of a crawler whose performance can be scaled up by adding additional low-cost processes and using them to run multiple crawls in parallel.

References

1. Bergman, M.K.: The deep web: Surfacing hidden value. J. Electron. Publ. **7**(1), 1174–1175 (2001)
2. Sherman, C., Price, G.: The Invisible Web: Uncovering Information Sources Search Engines Can't See. CyberAge Books, Medford (2001)
3. Raghavan, S., Garcia-Molina, H.: Crawling the hidden web. In: 27th International Conference on Very large databases (Rome, Italy, September 11–14: VLDB'01), pp. 129–138. Morgan Kaufmann Publishers Inc., San Francisco (2001)
4. Ntoulas, A., Zerfos, P., Cho, J.: Downloading textual hidden web content through keyword queries. In: 5th ACM/IEEE Joint Conference on Digital Libraries (Denver, USA, Jun 2005)

JCDL05, pp. 100–109 (2005)
5. Barbosa, L., Freire, J.: Siphoning hidden-web data through keyword-based interfaces. In: SBBD, 2004, Brasilia, Brazil, pp. 309–321 (2004)
6. Liddle, S.W., Embley, D.W., Scott, D.T., Yau, S.H.: Extracting data behind web forms. In: 28th VLDB Conference 2002, HongKong, China, pp. 38–49 (2002)
7. Chang, K.C.-C., He. B., Li. C., Patel, M., Zhang, Z.: Structured databases on the web: Observations and implications. SIGMOD Rec. *33*(3), 61–70 (2004)
8. Gupta, S., Bhatia, K.: Exploring 'hidden' parts of the web: The hidden web. In: Lecture notes in Electrical Engineering, Proceedings of the International Conference ArtCom 2012, pp. 508–515, Springer, Heidelberg (2012)
9. Gupta, S., Bhatia, K.: A system's approach towards domain identification of web pages. In: Proceedings of the Second IEEE International Conference on Parallel, Distributed and Grid Computing (India, December 6–8, 2012) PDGC'12, IEEE Xplore
10. Lawrence, S., Giles, C.L.: Accessibility of information on the web. Nature **400**, 107–109 (1999)
11. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval, 2nd edn. Addison-Wesley-Longman, Boston (1999)
12. Ipeirotis, P.G., Gravano, L., Sahami, M.: Probe, count, and classify: Categorizing hidden-web databases. In: Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 67–78, Santa Barbara, CA, USA, May (2001)
13. Wang, W., Meng, W., Yu, C.: Concept hierarchy based text database categorization. In: Proceedings of International WISE Conference, pp. 283–290, China, June (2000)
14. Chakrabarti, S., van den Berg, M., Dom, B.: Focused crawling: A new approach to topic-specific web resource discovery. In: Proceedings of the 8th International WWW Conference (1999)
15. Zhang, Z., He, B., Chang, K.C.-C.: Light-weight domain-based form assistant: Querying web databases on the fly. In: Proceedings of the 31st Very Large Data Bases Conference (2005)
16. McCallum, A., Nigam, K., Rennie, J., Seymore.K.: Building domain-specific search engines with machine learning techniques. In: Proceedings of the AAAI Spring Symposium on Intelligent Agents in Cyberspace (1999)
17. Chakrabarti, S., Punera, K., Subramanyam, M.: Accelerated focused crawling through online relevance feedback. In Proceedings of WWW, pp. 148–159 (2002)
18. Rennie, J., McCallum, A.: Using reinforcement learning to spider the web efficiently. In Proceedings of ICML, pp. 335–343 (1999)
19. Diligenti, M., Coetzee, F., Lawrence, S., Giles, C.L., Gori.M.: Focused crawling using context graphs. In: Proceedings of the 26th International Conference on Very Large Databases, pp. 527–534 (2000)
20. Profusion's search engine directory. http://www.profusion.com/nav