Surfacing Deep Web behind Scripting Languages

Bharat Bhushan[1] , Narender Kumar[2]

[1]Department of Computer Science & Applications, Guru Nanak Khalsa College, Yamuna Nagar (Haryana), India
[2]Research Scholar, Singhania University , V.P.O.-Pacheri Bari, Dist. Jhunjhunu, Rajasthan - [INDIA] under Enrollment No. 1050103206

Summary
This paper discusses The crawler can't crawl everything from open web as few of content which is hidden in websites in different forms like file formats (PDF, Flash, Office files, streaming media) or Dynamically generated pages (cgi, JavaScript, asp, or most pages with "?" in URL). And this is very challenges phase to crawl this invisible / hidden web/ deep web. Mostly the important / valuable information on web is in form of hidden and / or is dynamically generated by technologies such as java script. In this paper a technique is proposed to surface the information behind the scripting languages like java scripts.

Keywords :
Website, Dynamically generated pages, Deep Web , Java Scripts , Invisible Web, crawler

1. Introduction

The Deep Web is the largest growing category of new information on the Internet. Deep Web content is highly relevant to every information need, market, and domain. A great amount of valuable information is "hidden" behind the query forms of online databases, and/or is dynamically generated by technologies such as JavaScript. This portion of the web is usually known as the Deep Web or the Hidden Web.
Searching on the Internet today can be compared to dragging a net across the surface of the ocean. While a great deal may be caught in the net, there is still a wealth of information that is deep, and therefore, missed. In this Information Age , the value of deep Web content / information, is immeasurable The reason is simple: The crawler engines of today cannot reach most of the information contained in the Web. A full ninety-five per cent of the deep Web is publicly accessible information — not subject to fees or subscriptions but Most of the Web's information is buried far down on dynamically generated sites, and standard search engines never find it.[1]

2. Related Work

Manuel Álvarez et. al described the architecture of a crawling system able to access the contents of the hidden web.They focused on the techniques used to access the content behind web forms. Their approach is based on a set of domain definitions, each one describing a data-collecting task. From the domain definition, the system uses several heuristics, based on visual distance and text similarity measures, to automatically identifying relevant query forms and learning how to execute queries on them. [2]
Sriram Raghavan et.al designed a crawler capable of extracting content from hidden Web. They also present results from experiments conducted to test and validate techniques. They proposed an application/task specific approach to hidden Web crawling. A narrow application focus is also useful in designing a crawler that can benefit from knowledge of the particular application domain. They presented a simple operational model of a hidden Web crawler that succinctly describes the steps that a crawler must take, to process and submit forms. [3]
Jayant Madhavan et. al described the technical innovations underlying the first large-scale Deep-Web surfacing system. The results or surfacing are currently enjoyed by millions of users per day world-wide, and cover content in over 700 domains, over 50 languages, and from several million forms. The impact on their search tra□c is a significant validation of the value of Deep-Web content. They work illustrates three principles that can believe raged in further investigations. First, the test of in formativeness for a form input can be used as a basic building block for The Deep Web, i.e., content hidden behind HTML forms, has long been acknowledged as a significant gap in search engine coverage. Since it represents a large portion of the structured data on the Web, accessing Deep-Web content has been along-standing challenge for the database community.[4]
Ping Wu1 et. al focused on the central issue of enabling efficient Web database crawling through query selection, i.e. how to select good queries to rapidly harvest data records from Web databases. They model each structured Web database as a distinct attribute-value graph. Under this theoretical framework, the database crawling problem is transformed into a graph traversal one that follows "relational" links. They showed that finding an optimal query selection plan is equivalent to finding a Minimum Weighted Dominating Set of the corresponding database graph, a well-known NP-Complete problem. They proposed a suite of query selection techniques aiming at optimizing the query harvest rate. Extensive experimental

Manuscript received July 5, 2012
Manuscript revised July 20, 2012

evaluations over real Web sources and simulations over controlled database servers validate the effectiveness of their techniques and provide insights for future efforts in this direction [5]

J. Callan and M.Connell presented a new technique to automatically create a description (a sample) for the database. They argued that accurate description can be learned by sampling a text database with simple keyword-based queries. [6].

Anuradha and A.K.Sharma automatically detects the domain specific search interfaces by looking the domain word in the URLs, then title and after that attributes of the source code. Feature space model classified the web pages into a set of categories using domain ontology. The large-scale collections of query interfaces of the same domain are then integrated into integrated search interface (ISI). This interface will permit users to access information uniformly from multiple sources of a given domain. [7]

Ipeirotis P. and Gravano L presented an algorithm to derive content summaries from "uncooperative" databases by using "focused query probes," which adaptively zoom in on and extract documents that are representative of the topic coverage of the databases. Their content summaries are the first to include absolute document frequency estimates for the database words. They also presented a novel database selection algorithm that exploits both the extracted content summaries and a hierarchical classification of the databases, automatically derived during probing, to compensate for potentially incomplete content summaries. Finally, they evaluated their techniques thoroughly using a variety of databases, including 50 real web-accessible text databases. Their experiments indicated that new content-summary construction technique is efficient and produces more accurate summaries than those from previously proposed strategies. [8]

A. Ntoulas et. al studied how can one build an effective Hidden Web crawler that can autonomously discover and download pages from the Hidden Web. Since the only "entry point" to a Hidden Web site is a query interface, the main challenge that a Hidden Web crawler has to face is how to automatically generate meaningful queries to issue to the site. Here, They provided a theoretical framework to investigate the query generation problem for the Hidden Web and proposed effective policies for generating queries automatically. Their policies proceed iteratively, issuing a different query in every iteration. they experimentally evaluated the effectiveness of these policies on 4 real Hidden Web sites and their results are very promising. For instance, in one experiment, one of our policies downloaded more than 90% of a Hidden Web site (that contains 14 million documents) after issuing fewer than 100 queries. [9]

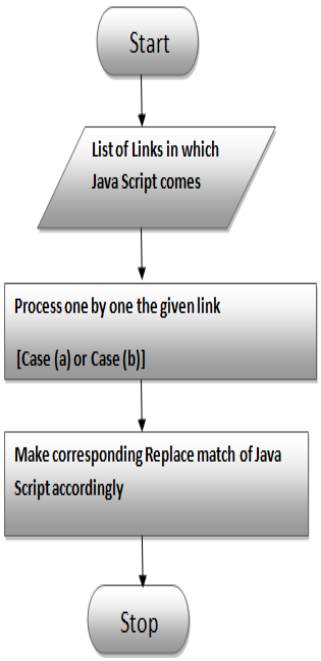
Liddle, S., et. al described domain independent approach for automatically retrieving the data behind a given Web form. They prototyped a synergistic tool mostly in Java, but also using JavaScript, PHP, and Perl that brings the user into the process when an automatic decision is hard to make. Here they used a two-phase approach to gathering data: first they sampled the responses from the Web site of interest, and then, if necessary, they methodically try all possible queries(until either they believe they have arrived at a fix point of retrieved data, or we have reached some other stopping threshold, or they have exhausted all possible queries).[10]

3. Proposed Model

Here the crawler extracts the links from a page & than crawls/downloads them also. But all links are not relative or absolute URLs rather few links calls javascript functions to determine the actual link. Browser handles that but not our crawler (i.e. Handling Javascript links) at its core. So we have a workaround to manually specify that on this domain convert the javascript links of this format to some other format. Here we are trying to put some text explaining how to define such match/replace for a crawler for some website , which has javascript links on it.

It will show the error if highest replace variable is greater than the number of sub patterns i.e. upto 11 digits numbers e.g. javascript:submitBID('29043808242');

Flow Chart :



Case Study

Case(A)

Problem : How to crawl the JavaScript Link from the Deep Web. It will be visible when we open the link

Sol. Here

1.
Link You want to crawl

2.
Find the javascript link from the page

3.Click on the javascript link & see what kind of link is generated when clicked

4.Understand some regular expression(Match/Replace) basics and make the corresponding link for crawl.

Example :

1. Link You want to crawl

A: http://www.xyz.com/Index.asp

2. Find the javascript link from the page

E1: javascript:openWindow('moreinfo.asp?reset=true&id=72', 750, 550)

3. Click on the javascript link & see what kind of link is generated when clicked

T: http://www.xyz.com/moreinfo.asp?reset=true&id=72

4. Understand some regular expression(Match/Replace) basics and make the corresponding link for crawl.

Technique:

We saw that in javascript links its just the 72, 75 ie ids which are varying in T.

OPEN T: Directly in browser & try to change the id=72 to 71 etc & see if it works & if works great.

* Now we have to write a regular expression pattern which can convert E1 to T

As the links can contain not only javascript links but any other well formed url also. So One has to specify a pattern by which the crawler can decide which links are the be replaced.

Call it MATCH. It would be something like this in our example

<Match><![CDATA[javascript:openWindow\('(.*?),.*\)]]></Match>

Now what should crawler do if such url is found. that need to be specified in another pattern. Call it REPLACE.

<Replace><![CDATA[http://http://www.xyz.com/$0]]></Replace>

Here we said that whatever you find in the section/group (.*?) which becomes $0 append that to some link,

which Together will make a valid Link T. Now this is what the crawler will crawl.

Case (B)

Problem : How to crawl the JavaScript Link from the Deep Web. which link are generated after opening the link/webpage.

Sol.

1.
Link You want to crawl

2.Some post parameters are identified using HTTPAnalyzer & LiveHTTPHeaders which helps us to find out the new response to be sent for crawling

3.Understand some regular expression(Match/Replace) basics and make the corresponding link for crawl

Example :

1. Link You want to crawl

Url A = Url C. ie Url remains the same when we click on javascript link.

2. Some post parameters are sent which identifies what new response to be sent.

For that one need to use some tool Like HTTPAnalyzer & LiveHTTPHeaders to see what POST PARAMETERS are sent and try sending them in the URL itself & see if it works. To my knowledge if it works you can create a match replace else crawler cannot handle. Crawler Handles

Post Parameter definition & control for depth=0 request only.

If possible you can break all such links into several XMLs.

3. You need to understand some regular expression(Match/Replace) basics

eg WildCard Characters use (like .?*)([]^+)

4. Discussion and Conclusion

First of all , one should identifying the approaches for searching the invisible web. It will give a clear idea of what is the goal /search & how one can access the relevant information from the deep web. Here in this paper we have tried to crawl the dynamically generated pages using two techniques get & post. We proposed the model to crawl hidden web where the general crawler not able to identify /crawl the hidden/Deep web. By using this technique , we can Surface the hidden information from open web. This paper will be an asset to retrieve the valuable & important information which one can't obtain by crawling the web.

References

[1] Bergman, Michael K. , "White Paper: The Deep Web: Surfacing Hidden" Volume 7, Issue 1, August, 2001 http://dx.doi.org/10.3998/3336451.0007.104

[2] Manuel Álvarez1, Juan Raposo1, Alberto Pan1*, Fidel Cacheda1, Fernando Bellas1, Víctor Carneiro1 "Crawling the Content Hidden Behind Web Forms 1" Proceeding ICCSA'07 Proceedings of the 2007 international conference on Computational science and Its applications - Volume Part II Pages 322-333 Department of Information and Communications Technologies,University of A Coruña,

[3] Sriram Raghavan Hector Garcia-Molina "Crawling the Hidden Web" Proceeding VLDB '01 Proceedings of the 27th International Conference on Very Large Data Bases San Francisco, CA, USA ©2001 Pages-129-138 ISBN:1-55860-804-4

[4] Jayant Madhavan David Ko Łucja Kot Vignesh Ganapathy Alex Rasmussen "Google's Deep-Web Crawl" Journal Proceedings of the VLDB Endowment VLDB Endowment Hompage archive Volume 1 Issue 2, Pages 1241-1252 August 2008

[5] Ping Wu1,Santa , Ji-Rong "Query Selection Techniques for Efficient Crawling of Structured Web Sources" Proceeding ICDE '06 Proceedings of the 22nd International Conference on Data Engineering IEEE Computer Society Washington, DC, USA Page-47 , 2006 ISBN:0-7695-2570-9

[6] J. Callan, M.Connell: "Query-based sampling of text databases". ACM Transactions on Information Systems , pages 97–130 . 2001

[7] Anuradha, A.K.Sharma, "A Novel Approach for Automatic Detection and Unification of Web Search Query Interfaces using Domain Ontology" International Journal of Information Technology and knowledge management(IJITKM), August 2009.

[8] Ipeirotis P., Gravano L. "Distributed Search over the Hidden Web: Hierarchical Database.Sampling and Selection." In Proceedings of the 28th International Conference on Very Large Databases. 2002.

[9] A. Ntoulas, P. Zerfos, and J. Cho. "Downloading Hidden Web Content." Technicalreport, UCLA,2004.

[10] Liddle, S., Embley, D., Scott, Del., Yau Ho, Sai. "Extracting Data Behind Web Forms." Proceedings of the 28th Intl. Conference on Very Large Databases. 2002.

Bharat Bhushan received the M Sc (Physics), from Panjab Univ. Chandigarh and M.Sc. (Comp. Sc.), MCA degrees from Guru Jambeshwar University. Respectively. Presently working as Head, Department of Computer Science and Applications, Guru Nanak Khalsa College, Yamuna Nagar (affiliated to Kurukshetra University, Kurukshetra-Haryana, India) and senior most teacher of computer science in Haryana since 1984. He is a member of Board of Studies of Computer Science, Kurukshetra University and member of Advisory Board of educational programme (EDUSAT) launched by Govt. of Haryana to impart online education. His research interest includes Software engineering, Digital electronics, networking and Simulation Experiments.

Narender Kumar received B.Sc. (Computer Science & Application) degree from Guru Nanak Khalsa College, Yamunanagar Affiliated to Kurukshetra University and MCA Degree from Ch. Devil Lal Post Graduate Regional Center of Kurukshetra University. He is the Research Scholar, Singhania University , V.P.O. - Pacheri Bari, Dist. Jhunjhunu, Rajasthan - [INDIA] under Enrollment No. 1050103206.