Volume 3, Issue 4, April 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Prefetching Data from Hidden Web with DSIM Architechture

Babita Saharawat*
Department of CSE & M.D.U university
at Manav Rachna college of Engineering

Ashok Goyal
Department of IT
Manav Rachna college of Engineering

Abstract— The paper explore the problems identification, objectives, architecture of prefetching of hidden web pages using (DSIM). In this paper, a novel framework for interface matching is being proposed. Large amount of on-line information inside on the hidden web (deep or invisible web) that is not accessed by every user. These web pages are generated dynamically from databases demand on user query. These such pages not indexed by a static URL, it generated only when a user fire query and result displayed result. Now a day's many different matching solutions have been proposed so far. The rapid growth in the amount of information and the number of users has led to difficulty in providing effective search services for the web users and increased web latency; resulting in decreased web performance. Most of the search engines deal with surface Web only, the set of Web pages directly accessible through hyperlinks, mostly ignoring the vast amount of information hidden behind forms, which composes by the hidden Web. As compared to the Surface Web, the hidden Web contains a much larger amount of high-quality information hidden behind the databases. This framework extracts hidden web pages by accruing benefits of its unique features: 1) automatic downloading of matching keyword and prefetching related data from hidden web databases, 2) identification of fuzzy mappings between search interface elements by using a novel approach called DSIM (Domain-specific Interface Mapper), and 3) the capability to automatic filling of search interfaces. The effectiveness of proposed framework has been evaluated through experiments using real web sites and encouraging preliminary results were obtained.

Keywords— DSIM, Hidden web, Search engine, Fuzzy matching and Spiders.

I. INTRODUCTION

The Internet, sometimes called simple "Net". Net is a worldwide system of computer networks – a network of networks in which users at any one computer can get information from any other computers or any other computer based communication device. A multitude of search engines, such as Google, Yahoo, and AltaVista etc., are available to retrieve information from the huge repository of hyperlinked information called World Wide Web. WWW or Web is a system of interlinked collection of billions of documents and millions of users formatted using HTML. A user views web pages that may contains text, images, and other multimedia documents by hyperlinks. The information is hidden behind in the form of HTML forms and it only available on response to user's request [4]. It is estimated that there are several million hidden-web sites. Web pre-fetching becomes an important solution where in forthcoming page accesses of a client are predicted, based on domain 1information. This dissertation will propose an approach for increasing web performance by analysing user domain and perfecting the frequently accessed pages after completing the web structure, so as to provide relevant information to the user. As the web is vast resources of information, how to find just the right bit of information that user need or how to provide relevant information to the user from the internet with in a limited time is a big challenge in information retrieval. Hidden web is used for data extraction from web with the help of search engine. Web information can't access without the search engine. Currently used search engines can't make index to the pages which are generated automatically by the back end databases called deep web. Large amount of on-line information is hidden on the invisible web (deep or hidden web). These web pages are generated dynamically from databases and other data sources hidden from the user, these pages are not indexed by a static URL. These pages are generated only when queries are asked via a search interface rendering interface matching a critical problem in many application domain. These engines crawl the web by following URLs that are embedded in the Web pages. The downloaded Web pages are stored and indexed into the local databases of the search engines. When a request in the form of keywords is arrives, then the local databases are searched and gives the appropriate web pages are returned to the user.

Information searching on web has becoming one of the most important and popular activities. As the Web grows faster, more and more data has become available on demand of user and database accessed in the form of HTML. With the popularization of the WWW, a huge amount of data from a number of different domains has become available on user demand. These search engines only deal with the surface Web, the set of Web pages directly accessible through hyperlinks, mostly ignoring the vast amount of information hidden behind forms, which composes the hidden Web (also known as deep Web or invisible Web). The hidden Web contains a much larger amount of high-quality information hidden behind the databases.

II. PROBLEMS IDENTIFICATION

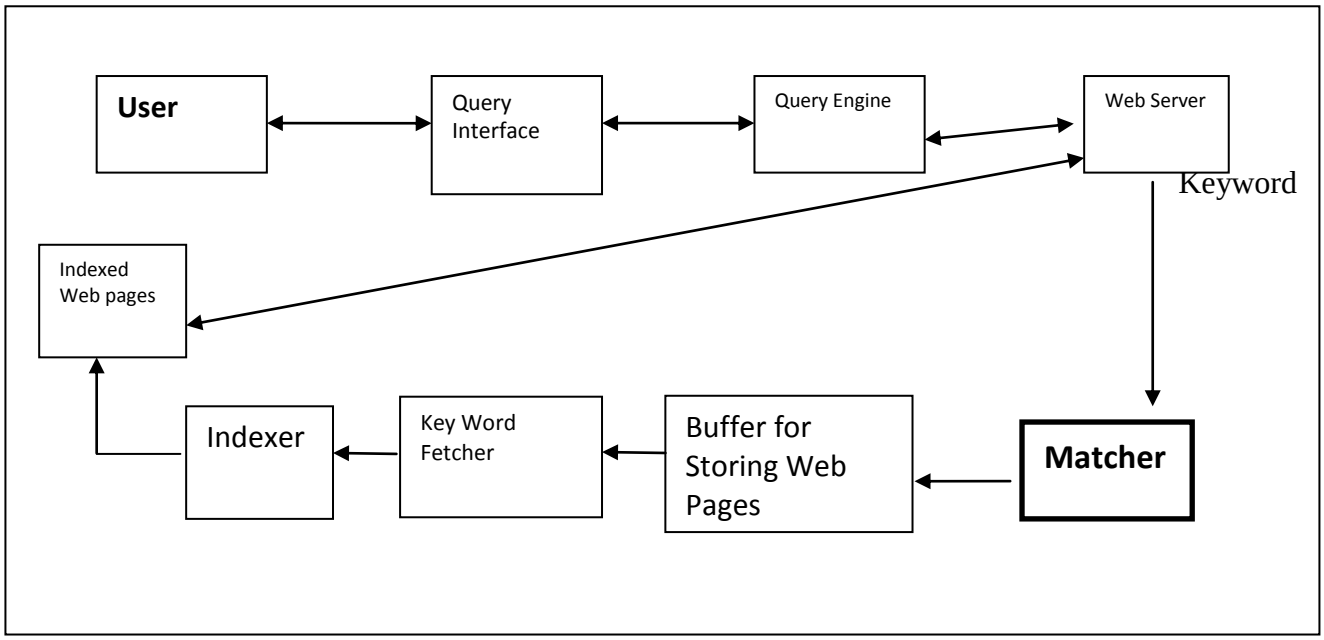
PROBLEM FORMULATION

- In the normal working of search engine, the search engine returns millions of web pages as related to user query result but all the pages search engine first search's its own local database and if there the desired web pages doesn't found then it fetches from www.
- Though all the pages prefetched by the search engine are not related to our work and it is not possible to view all the pages by clicking over the link. Fetching technique is fetch more data from hidden web and stored in local server that means any user search start with finding keyboard but fetched technique also searched and fetched data for mouse and monitor with the keyboard.
- In case of hidden web where the size is more larger than the surface web this would be a difficult task to handle more web pages and store in local database.

OBJECTIVES OF THE STUDY

Our objective is to make a method called prefetching data with the help of DSIM, where data is fetched from www or local database. The search engine return the most relevant pages on the top of the list because it is a normal tendency of searching but data is PREFETCHING from Hidden Web Pages Using (DSIM) with less timing for searching.

III. PROPOSED ARCHITECTURE OF PREFETCHING OF HIDDEN WEB PAGES USING (DSIM)

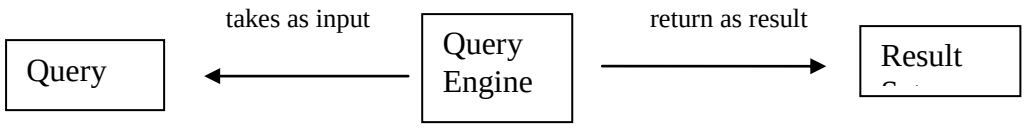


Query Interface: - Query Interface is a mechanism in COM (Microsoft's Component Object Model) for determining a known component supports a specific interface. Anyone can use the current interface pointer to point the interface and result interface ID passing, and interface ID moves one position to another position. If the object supports an interface which matches the ID, it gives reply back a pointer that can told about the matching result of user query. Pointer always point the correct type matching and gives matching result query interface ID.

Query Engine: -Query Engine is a service that takes a description of a search request, evaluates and executes the user request, and returns the results back to the user. Query Engine acts as an intermediate layer between the clients and the underlying data sources by interpreting search requests and shielding the clients from details on how to access the data sources. A Query Engine separates the formulation of individual search requests from their execution. It encapsulates the process how data sources are accessed, how native query statements are formulated, and how those statements are executed. Queries may be created at runtime if they are based on dynamic user input or they may be created at compile-time if they are based on static business requirements.

Query engine three participants:-

1) Query: - A Query describes a search request. It includes the search target, a condition that refers to the data records to be searched, as well as additional options.
2) Query Engine: - The Query Engine performs a search request: it translates a Query into a native query statement, executes it on a data source, and returns a Result Set.
3) Result set: - A Result Set encapsulates the results retrieved by the Query Engine.



Web server: -Web servers are computer that *serves up (*deliver) web pages. A Web server is a basically a program that, using the both sides (client/server site) model and the World Wide Web's and used Hypertext Transfer Protocol(HTTP). Web serves files that form Web pages to web users (whose computers contain HTTP clients that forward their requests). Every web server has an own IP Addresses and domain name worldwide. Any computer can be turned into a Web server by installing server software and connecting the machine to the Internet. Every computer on the Internet that contains a web site must have a web server program. There are two leading web servers that are Apache server, the most widely-installed Web server, and Microsoft's Internet Information Server (IIS). Web sites are always used for search the documents based on the user given keywords or query. For searching web server need a Search engines. Search engine typically search millions of web sites who saved text in the form of web pages. Search engine or a web search engine is a software code that is designed to search information on the WWW. Search engine used "Spiders", which search the web information. The search results are generally presented in a line of results and referred to as search engine results pages. The information may be in the form of web pages, images, videos, pdfs and other types of files. More valuable information is stored the hidden databases at one single location.

Buffer: -Buffer is a temporary storage area, that usually in the RAM. The purpose of the buffer is to holding some area in the RAM for processing, only for some time (processing time only). Buffer is used to manipulate the data before transferring it to a device. Because the processes of reading and writing data to a disk are relatively slow, many programs keep track of data changes in a buffer and then the buffer into a disk. When you save the file, the word processor updates the disk file with the contents of the buffer. Buffer is much more efficient than accessing the file on the disk each time, it also make a change to the file.

There are many benefits to gain by placing these systems:

- Saving working hours
- An operator can support multiple processes
- Contributing to a continuous production

Key word fetcher: -Keyword is the percentage of times a keyword or a phrase appears on a web page compared to the total number of words on the page. Choosing the Right Keywords to Optimize For Keyword

Keywords special Places are

• Keywords in URLs and File Names

• Keywords in Page Titles

• Keywords in Headings

(User must require some knowledge of HTML Coding).

Fetcher always fetch (or downloads) all old data matching results as well as new data matching data from the database. Fetcher has very large amount of data to fetch from web, fetching can be started by a single keyword or a string. Fetcher fetch the related keyword data from database and gives matching keyword data to the user within few nanoseconds.

Matcher: - DSIM (Domain Specific Interface Mapper) is an ideal technique for searching the resources according to the domain. The access to the deep web will gives the information retrieved and relevant. The interface matching technique helps in discovering the necessary attributes across the Web Interfaces. **DSIM** basically consists of the three phases: -

1) **Parsing**
2) **Matching**
3) **Mapping Generation**

In the proposed work the **Matching** is used for increasing the web performance by analysing DSIM technology and perfecting the frequently accessed pages and provide relevant information to the user with**i**n few seconds with the help of FUZZY MATCHING.

Matching: - Semantic Matching phase of DSIM Matching the components of two different interfaces by using a domain specific matching. It uses three types of matching strategies that are

1) **Fuzzy Matching**
2) **Domain-specific Thesaurus**
3) **Data type matching**

Fuzzy Matching: - This Fuzzy matching uses a single element matcher that is called Node Name Matcher (NNM). The NNM can be implemented using the **CompareStringFuzzy** Function. The **CompareStringFuzzy** Function compares two strings and returns a similarity value in the range [0, 1].

The retrieving data from hidden web sites has two tasks:

1) Resource Discovery
2) Content Extraction

Resource Discovery: - The task deals with the automatically finding the relevant Web sites containing the hidden information. Use the sets of relevant Web pages as starting points. To find the relevant Web pages, domain related words [4] were submitted to the search engine, and checked the domain specific forms [5]. In this process few categories of forms were discarded which are irrelevant to hidden databases. Such as, form which has any password HTML type element, if form action is JavaScript Action, personnel information filling forms and forms whose weight is less than the threshold value.

Content Extraction: - The task deals with obtaining the information from those sites by filling out forms with relevant keywords. The system considers only the HTML form as an ordered list of values such has URL, form the name, form the method, form the action, form the id, form the elements and form weight. Then all the HTML form values are extracted from the relevant forms and sent to the database.

Indexer: - After a page is crawled on web, the next step is to index the page (content). The indexed page is stored in a local database (giant database), from where it can later be retrieved. The process of indexing the pages, is identifying by the words and expressions that best describe the page and assigning the page to particular keywords. Indexer is very popular now a day because it arrange the whole web data one by one web pages on local database server.

IV. CONCLUSION AND FUTURE SCOPE:

In a conclusion we can say that DSIM quickly identifies the regions in the interface repository comprising of important fuzzy mappings. The tests conducted on Domain-specific Hidden Web DSIM indicate that it efficiently search the hidden web pages. It further improves by discarding the less important mappings as fuzzy mapping uses a comparestringfuzzy as a selection parameter. The loss mostly occurs among the mappings which rank low, an acceptable trade off. The domain-specific Interface Matching Library currently supports three matching strategies, but it is extensible in the sense that newer and better strategies that is (1) Parsing, (2) Matching and (3) Mapping Generation also Future research includes: (1) Establishing a tighter control over selection the important mappings by selection parameters on the efficiency/effectiveness allows for better tuning in search, (2) ordering of the mappings in the Mapping Knowledge Base – a measure of mapping's quality can be used to decide which mappings have better chances to produce good mappings. In this way, the time-to-first good mapping can be improved, (3) extending the match library and improving the learning capability. Further work can be done in developing a specialized search engine for Hidden Web.

REFERENCES

[1] Bhatia, Komal Kumar and Sharma, A.K (2008) A Framework for Domain-Specific Interface Mapper (DSIM), IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.12, December.

[2] Khetwat, Saritha and Dharavath, Kishan (2011) Domain and Keyword Specific Data Extraction from Invisible Web Databases, Eighth International Conference on Information Technology: New Generations.

[3] "Anatomy of a Large-Scale Hyper textual Web Search Engine", Sergey Brin and Lawrence Page, Stanford,CA 94305,

[4] BrightPlanet. Com, The deep Web: Surfacing hidden value.

WEBGROPLY

1) http://www.en.wikipedia.org/wiki/Deep_Web retrieve on 22.04.2013
2) http://infolab.stanford.edu/~backup/google.html retrieve on 22.04.2013.
3) http://www.ijcaonline.org/volume15/number4/pxc3872579.pdf retrieve on 22.04.2013
4) http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5945400 retrieve on 22.04.2013
5) http://brightplanet.com.

