IOSR Journal of Engineering
Apr. 2012, Vol. 2(4) pp: 858-861

Hidden Web Indexing Using HDDI Framework

Shashank Agarwal[1], Saurabh Kaushik[1], Siddhartha Goel[1],
Mukunj Goel[1], Yogesh kumar Meena[1], Priyanka Gupta[2]

[1]*Department of Computer Science, Ajay Kumar Garg Engineering College, Gautama Budh Technical University,Ghaziabad*
[2] *Priyanka Gupta is the mentor, Department of Computer Science, Ajay Kumar Garg Engineering College and Gautama Budh, Technical University, Ghaziabad*

There are various methods of indexing the hidden web database like novel indexing, distributed indexing or indexing using map reduce framework. Our goal is to find an optimized indexing technique keeping in mind the various factors like searching, distribute database, updating of web, etc. Here, we propose an optimized method for indexing the hidden web database. This research uses Hierarchical Distributed Dynamic Indexing (HDDI) Framework for indexing the Data downloaded by the Siphone++ crawler. As HDDI technology develops, we are discovering novel approaches that address several issues of managing distributed digital information within the context of the HDDI paradigm.

1.1 Proposed Work

In this Paper, we propose an optimized method for indexing the hidden web database. The basic model is depicted in Figure 1.1. Section 1.2 discusses first component Siphone++ as a hidden web crawler. It is the advance version of Siphone with new framework for Indexing. Section 1.2.1 discusses the Adaptive component of Siphone++. Section 1.2.2 outlines the Heuristic approach of Siphone++ to siphon the data behind the Search Interfaces. It issues the queries that have high coverage of data from hidden database.

Section 1.3 proposes HDDI framework for indexing the hidden web Data. HDDI is a novel approach to organizing large quantities of unstructured data in a loosely coupled distributed environment underdevelopment at Lehigh University and at the National Center for Supercomputing Applications. The approach is based on the algorithmic creation of subtopic regions of semantic locality in sets of distributed documents; this allows automatic discovery of similarities at a fine level of granularity amongst concepts within documents. In this way, hierarchical indices (such as those created now "by hand" in many places on the web; www.yahoo.com is probably the most well-known example) are generated for topics in documents in a volatile, distributed environment, providing the information seeker with an always up-to-date map of information spaces. The ability to generate large hierarchical indices on the fly allows for a realistic, useful mapping of cyberspace without the need for time-consuming human intervention.

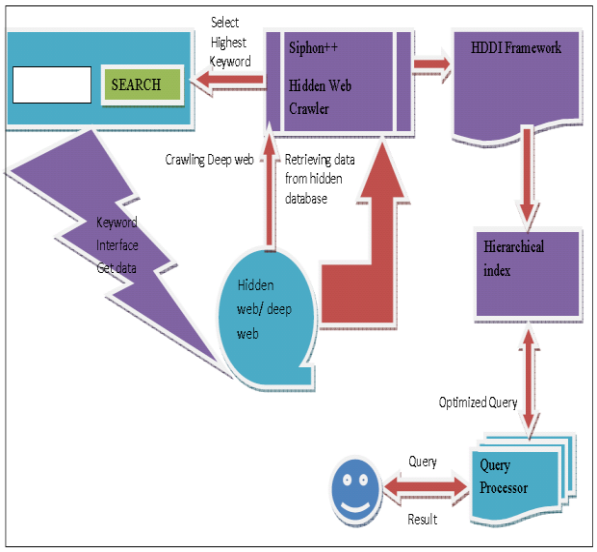

Fig. 1.1: Proposed Optimized Model for indexing

1.2 Using SIPHONE++ as Hidden Web Crawler

Reference [1] presents a crawler for automatically retrieving Web content hidden behind keyword-based interfaces. Being able to retrieve and index, this content has the potential to uncover hidden information and help users find useful information that is currently out-of-reach for search engines. Unlike multi-attribute forms, keyword-based interfaces are simple to query, since they do not require detailed knowledge of the schema or structure of the underlying data.

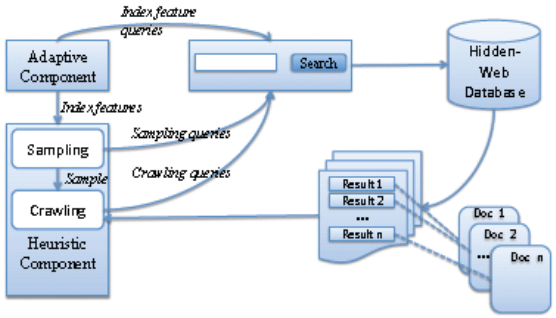

Fig. 1.2: Architecture of the Siphon++

It is thus possible to create automatic and effective solutions to crawl these interfaces.

1.2.1 Adaptive Component

The Adaptive Component (AC) detects the index features by issuing probe queries against the search interface.We call these queries index feature queries (see Figure 1.2).Two techniques commonly used for compacting search engine indexes are: stop word removal (*e.g.*, the removal of prepositions and articles); and stemming, *i.e.*, reducing words to their root (stem) form.

1.2.2 The Heuristic Component

The Heuristic Component is responsible for defining a policy for submitting queries. Two phases are:-

Sampling

The goal of the Sampling phase is to assemble are representative sample of the database. Reference[2] developed sampling-based algorithm that automatically discovers keywords which result in high recall; and use these keywords to build queries that siphon all available results in the database (or, as many as possible).

Crawling

After sampling, it selects most frequent words in documents in this sample to crawl the database, assuming they also have a high frequency in the actual database/index. The Crawling phase is responsible for

(1) Issuing queries to the database;

(2) Retrieving the result pages and extracting from them the links to the target documents; and

(3) Downloading the documents from the database.

1.3 Using HDDI Framework for Indexing

Why it is needed?

Traditional methods of indexing combine multiple subject areas into a single, monolithic index. There are enough documents on the Web that such indexing technology often fails to perform effective search. The difficulty lies in the fact that since so many documents and subjects are being combined together, retrieving all the documents that match a particular word phrase often returns too many documents for effective use. This problem has been known for some time [3]. Solutions based on link analysis have mitigated these difficulties yet fail to adequately address issues of recall [4].

In order to properly address this problem, a paradigm shift is needed in the approach to indexing. First and foremost, it is clear that digital collections are now and will continue to be distributed. Our first premise is thus that *indices* must also be distributed.

Secondly, it must be realized that the information contained in these distributed digital repositories can be classified in a hierarchical manner. Our second premise is thus that distributed indices must properly reflect the hierarchical nature of knowledge.

Thirdly, due to the vast increase in communications bandwidth and computing and online storage capabilities mentioned above, digital collections are frequently updated. This process reflects a key characteristic of 21st century collections:namely, they are dynamic in nature. Our third premise is thus that any new information infrastructure must include *dynamic* indexing capabilities.

1.3.1 Process of building a hierarchical index

Figure 1.3 depicts the steps involved in building a hierarchical index: concept identification/extraction; concept co-occurrence matrix formation; hierarchy construction; knowledge base creation; identification of regions of semantic locality; and hierarchy mapping.

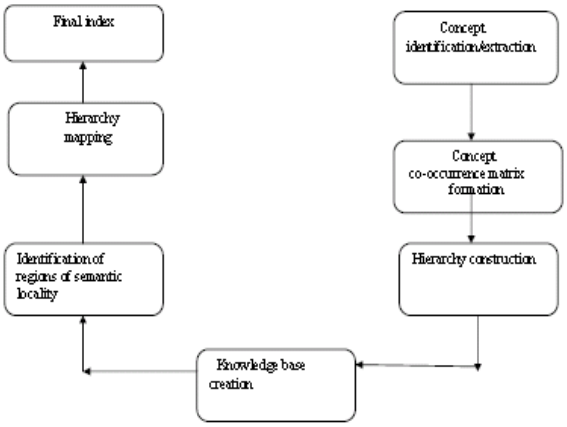

Fig.1.3 Process of building a hierarchical index

The parsing stage takes SGML, HTML or generalized XML tagged items as input. Based on AI techniques, our part of speech tagging approach includes the use of both lexical and contextual rules for identifying various parts of speech.

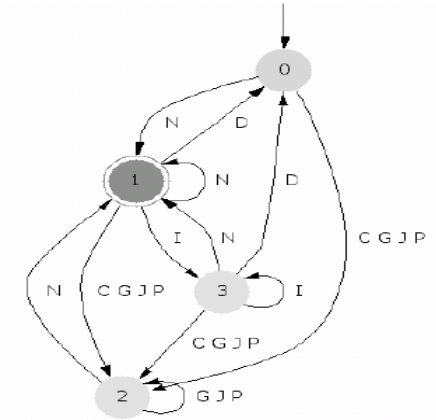

Figure 1.4 **A Finite State Automaton for Recognizing Complex Noun Phrases State** 0 is the start state, and state 1 the final state. C is a cardinal number,G a verb (gerund or present participle), P a verb(past participle), J an adjective, N a noun, I apreposition and D a determiner.

After identifying each word's part of speech, we invoke a finite-state machine, Figure 1.4 that accepts maximal length English-language noun phrases. [5], [6].

The final result of these steps is a reformulation of the original collection that includes a summation of the location and number of occurrences of each extracted concept. Co-occurring defines concepts that occur within the same item. The co-occurrence relation is reflexive and symmetric but not transitive. Given concepts extracted by the above process, we compute concept frequency and co-occurrence matrices.

Hierarchy construction is a meta-level organizational process that combines the matrices formed in the previous step. These co-occurrence matrices provide the basis for organizing concepts into ontology of knowledge based on the content of the collections.The resulting hierarchical index consists of high-resolution leaf-level index nodes that become increasingly less precise (i.e., more general) as the hierarchy is built.

Knowledge base creation is the second meta-level organizational process. For each matrix in the hierarchy constructed in the previous step, and for each concept in each matrix we compute a similarity with other concepts. This one-to-many mapping associates each concept with a list of related concepts ranked by similarity. More general concepts occur lower in the list. Each concept pair is weighted, creating asymmetric measures of pair-wise similarity between concepts. The similarity is a mapping from one concept to another that quantitatively determines how similar they are semantically. We term the resultant mapping a *knowledge base* .A knowledge base is represented as an asymmetric directed graph in which nodes are concepts and arc weights are similarity measures. The model building techniques are based on a *cluster function* [7].

The result is a knowledge base consisting of regions of high-density clusters of concepts. These regions consist of clusters of concepts that commonly appear together and collectively create a *knowledge neighborhood*. The computational core of sLoc is based on an algorithm due to Tarjan [8].

sLoc Algorithm:-It is used to detect strongly connected clusters. The figure below depicts the three steps of the sLoc process.

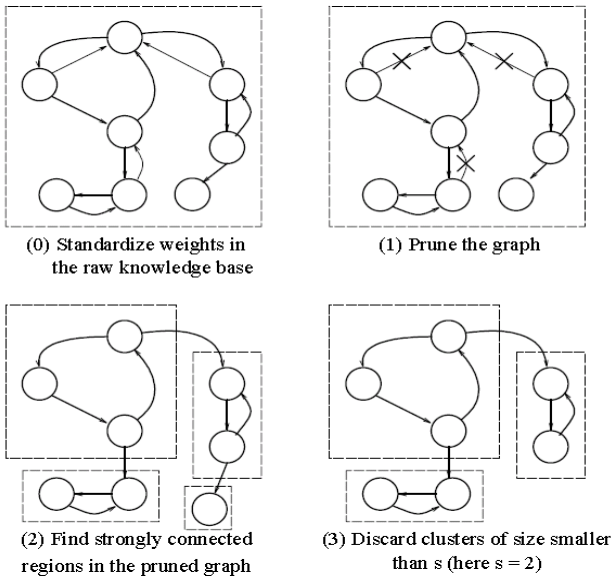

Figure 1.5 sLoc Algorithm

The final process is to create a mapping of the information space by developing linkages between the clusters. This process defines relationships between clusters, so that clusters are interconnected both with the child clusters that they were merged from and the higher-level parent clusters they were used to create. Upon completion of this phase, each level of the hierarchical index contains sufficient information to determine its relative position as a knowledge area in the overall knowledge hierarchy. This is an area of ongoing research [9].

Proposed Algorithm:-

Step-1 input item (html document) parsing
Step-2 part of speech tagging
Step-3 concept identification (uses finite state machine)
Step-4 compute concept frequency and co-occurrence matrices
Step-5 Systematic filtering
Step-6 pruning
Step-7 meshing lower level (child) matrices
Step-8 knowledge base creation {

Cluster function (){

$$ClusterWeight(C_j, C_k) = \frac{\sum_{i=1}^{n} d_{ijk}}{\sum_{i=1}^{n} d_{ij}} \times WeightingFactor(C_k)$$

wher
e

$d_{ij} = tf_{ij} \times \log\left(\frac{N}{df_j} \times w_j\right)$ product of concept frequency and inverse document frequency

$d_{ijk} = tf_{ijk} \times \log\left(\frac{N}{df_{jk}} \times w_j\right)$ combined weights of both concepts C_j and C_k in document i

}

}

Step-10 identify regions of semantic locality {

let N be the total number of nodes

let A the total number of arcs

An arc aij € A connects node ito node j

Let W be the set of arc weights in the graph

$w_{i,j}$ the weight of the arc going from node ito node j.

Therefore $W = \{w_{i,j}\}_{(i,j) \in \square N^2}$.

Mean of arc weights $M=1/A \sum_{2}^{(I,i)\in N} w_{i,j}$

Standard Deviation $SD = \sqrt{1/(A-1) \sum_{(i,j)\in N^2} (w_{i,j} - M)^2}$

Use mean and SD to compute threshold τ which is used to prune the graph}

Step-10 hierarchy mapping

Step-11 final hierarchical index

1.4 Conclusion and Future Scope

In this paper we examined the problem of siphoning data hidden behind keyword-based search interfaces. We index the downloaded data by using HDDI framework. We proposed a pretty complex and completely automated strategy that can be quite effective in practice, leading to high coverage of data.

Today this model is being used in various applications. An area of HDDI™ research involves processing data that records both synchronous and asynchronous interactions between individuals. A related research initiative involves the design of a multimedia framework for constructive, inquiry-based learning for introductory and upper level computer science courses. The content is drawn from the field of Object-Oriented Software Engineering (OOSE). This National Science Foundation project includes development of HDDI textual data mining techniques for establishing temporal context that will be automatically employed by learnersto detect emerging trends in OOSE research [10].

We have outlined a set of core algorithms for building models in HDD™,a framework for mining and management of distributed textual data. Ongoing work continues to focus on the validation of HDDI™ model building and indexing techniques in a variety of applications.

1.5 Reference

[1] Karane Vieira, Luciano Barbosa, Juliana Freire, Altigran Silva, Siphon++: A Hidden-Web Crawler for Keyword-Based Interfaces. *Proc.ACM*, California, USA,pp.978-991,October 2010.

[2] Luciano Barbosa, Juliana Freire, "Siphoning Hidden-Web Data through Keyword-Based Interfaces", *Proc. of Journalof Information and Data Management*, pp 133-144, February2010.

[3] National Research Council, c*omputing the Future: A Broader Agenda for Computer Science and Engineering*, National Academy Press.

[4] S. Brin and L. Page, The Anatomy of a Large-Scale Hyper textual Web Search Engine, *Proceedings of the Seventh International World Wide Web Conference*, Brisbane, Australia, April.

[5] R. Bader, M. Callahan, D. Grim, J. Krause, N. Miller and William M. Pottenger, The Role of the HDDI™ Collection Builder in Hierarchical Distributed Dynamic Indexing, *Proceedings of the Textmine .01 Workshop, First SIAM International Conference on Data Mining*, April.

[6] L. Karttunen, Directed Replacement. *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics,* Santa Cruz, California.

[7] H. Chen, J. Martinez, T. Ng and B. R. Schatz, A Concept Space Approach to Addressing the vocabulary Problem in Scientific Information Retrieval: An Experiment on the Worm Community System, *Journal of the American Society for Information Science*, Volume 48, Number 1, January.

[8] R. E. Tarjan, Depth first search and linear graph algorithms, *SIAM J. Computing*, 1:146-160.

[9] Y. B. Kim, The Role of Hierarchical Models in Hierarchical Distributed Dynamic Indexing, M.S. Thesis, Department of Computer Science at the University of Illinois at Urbana- Champaign, June.

[10] G. Blank, William M. Pottenger, G. D. Kessler, The CIMEL Project: Constructive, Collaborative, Inquiry-based Multimedia E-Learning, http://www.eecs.lehigh.edu/~cimel.